Detecting and Jamming Dynamic Communication Networks in Ati-Access Environments
Project # FA9550-08-1-0190

PI: Prof. Stanislav Uryasev
CoPI: Prof. Panos Pardalos

March, 2011

20120918154

Summary

Three chapters present methods for handling various optimization problems related to sensors networks. The first chapter investigates mathematical programming techniques for solving a class of multi-sensor scheduling problems. We conducted several case studies and investigated the performance of robust optimization solvers for considered 0-1 linear programming problems. The second chapter presents a survey describing recent developments in the area of mathematical programming techniques for various types of sensor network applications. The corresponding mathematical programming formulations and solving methods are discussed. The third chapter considers several classes of problems that deal with optimizing the performance of dynamic sensor networks used for area surveillance under uncertainty with robust radio connectivity constraints. Various formulations of optimization problems and computational results are presented.

Introduction

The first chapter presents mathematical programming techniques for solving a class of multi-sensor scheduling problems. Robust optimization problems are formulated for both deterministic and stochastic cases using linear 0-1 programming techniques. Equivalent formulations are developed in terms of cardinality constraints. We conducted numerical case studies and analyzed the performance of optimization solvers for the considered problem instances.

The second chapter presents a survey describing recent developments in the area of mathematical programming techniques for various types of sensor network applications. We discussed mathematical programming formulations associated with these applications, as well as methods for solving the corresponding problems. We also addressed some of the challenges arising in this area, including both conceptual and computational aspects.

The third chapter considers several classes of problems that deal with optimizing the performance of dynamic sensor networks used for area surveillance, in particular, in the presence of uncertainty. The overall efficiency of a sensor network is addressed by minimizing the overall information losses, as well as ensuring that all nodes in a network form a robust connectivity pattern at every time moment (sensors communicate and exchange information in uncertain and adverse environments). The considered problems are solved using mathematical programming techniques that incorporate quantitative risk measures, which allow one to minimize or bound the losses associated with potential risks. The issue of robust connectivity is addressed by imposing explicit restrictions on the shortest path length between all pairs of sensors and on the number of connections for each sensor (i.e., node degrees) in a network. Formulations of linear 0-1 optimization problems and the corresponding computational results are presented.

Chapter 1

Robust Multi-Sensor Scheduling for Multi-Site Surveillance problem

1.1 Introduction

The task of area surveillance is important in a variety of applications in both military and civilian settings. One of the main challenges that one needs to address in these problems is the fact that the number of locations (sites) that need to be visited to gather potentially valuable information is often much larger than the number of available surveillance devices (sensors) that are used for collecting information. Under these conditions, one needs to develop efficient schedules for all the available sensors (that can be installed, for instance, on Unmanned Air Vehicles) such that the amount of valuable information collected by the sensors is maximized. One can formulate this problem in terms of minimizing the information losses associated with the fact that some locations are not under surveillance at certain time moments. In these settings, the information losses can be quantified as both *fixed* and *variable* losses, where fixed losses would occur when a given site is simply not under surveillance at some time moment, while variable losses would depend on how long a site has not been previously visited by a sensor. In particular, taking into account variable losses of information is critical in the case of strategically important sites that need to be monitored as closely as possible.

In addition, the parameters that quantify fixed and variable information losses are in many cases *uncertain* by nature. In previous related works in this area, the uncertainties in these parameters were not explicitly taken into account (see, e.g., [46]); however, the development of efficient techniques to minimize or restrict the information losses under uncertainty is the task of crucial importance. This paper proposes mathematical programming formulations that allow quantifying and restricting the risks of worst-case losses associated with uncertain parameters.

The mathematical programming formulations are first developed for the deterministic case, and the natural extensions of these formulations to the stochastic case (with uncertain information loss parameters) is made by utilizing quantitative risk measures allowing to control the conservativeness of the optimal strategy. In particular, the statistical concept referred to as Conditional Value-at-Risk (CVaR) is used in the proposed problem formulations under uncertainty. Using these techniques allows one to efficiently incorporate uncertainties in the considered optimization problems, as well as provides the means of balancing between the optimality and the robustness of the solutions. Equivalent reformulations and extensions of the considered problems are also presented.

We have also conducted numerical case studies to test the performance of the suggested algorithms. Since the considered problems are NP-hard and involve uncertain parameters, near-optimal solutions were found for large problem instances. It turned out that solving cardinality-based reformulations of the considered problems provided good quality solutions in a reasonable time.

1.2 Deterministic approach

This section introduces a general mathematical framework for multi-sensor scheduling problems. Initially, we utilize the concepts introduced in [46] that was developed for a deterministic case of a single-sensor scheduling problem. We then generalize and extend these formulations to the more realistic cases of multi-sensor scheduling problems, including the setups in uncertain environments. Assume that there are m sensors and n sites that need to be observed at every discrete time moment $t = 1, \dots, T$. We assume that a sensor can observe only one site at one point of time and can immediately switch to another site at the next time moment. Since m is usually significantly smaller than n, we have breaches in surveillance that can cause losses of potentially valuable information. Our goal is to build a strategy that optimizes a potential loss that is associated with not observing certain sites at some time moments.

We then introduce binary decision variables

$$x_{i,t} = \begin{cases} 1, & \text{if } i\text{-th site is observed at time } t, \\ 0, & \text{otherwise,} \end{cases} \tag{1.1}$$

and integer variables $y_{i,t}$ that denote the last time site i was visited as of the end of time t, $i = 1, \dots, n$, $t = 1, \dots, T$, $m < n$.

We associate a fixed penalty a_i with each site i and a variable penalty b_i of information loss. If a sensor is away from site i at time point t, the fixed penalty a_i is incurred. Moreover, the variable penalty b_i is proportional to the time interval when the site is not observed. We assume that the variable penalty rate can be dynamic; therefore, the values of b_i may be different at each time interval. Thus the loss at time t associated with site i is

$$a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t}). \tag{1.2}$$

In the considered setup, we want to minimize the *maximum penalty* over all time points t and sites i

$$\max_{i,t}\{a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})\}. \tag{1.3}$$

Furthermore, $x_{i,t}$ and $y_{i,t}$ are related via the following set of constraints. No more than m sensors are used at each time point; therefore

$$\sum_{i=1}^{n} x_{i,t} \le m, \ \forall t = 1, \dots, T. \tag{1.4}$$

Time $y_{i,t}$ is equal to the time when the site i was last visited by a sensor by time t. This condition is set by the following constraints:

$$0 \le y_{i,t} - y_{i,t-1} \le t x_{i,t}, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, \tag{1.5}$$

$$t x_{i,t} \le y_{i,t} \le t, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, \tag{1.6}$$

Note that the above constraints automatically ensure that the feasible values of $y_{i,t}$ are integer. It is easy to verify by considering possible values of binary variables $x_{i,t}$. Therefore, in the following mathematical programming problems, we can set the variables $y_{i,t} \in I\!R$, and the inclusion of these constraints will make these variables integer in any feasible solution. This enables us to decrease the number of integer (binary) variables in the considered problems.

Further, using the notation $C = \max_{i,t}\{a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})\}$ and standard linearization techniques, we can formulate the multi-sensor scheduling optimization problem in the deterministic setup as the following mixed integer linear program:

$$\min C \tag{1.7}$$

$$\text{s.t.} \quad C \geq a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t}),\ \forall i = 1, \ldots, n,\ \forall t = 1, \ldots, T, \tag{1.8}$$

$$\sum_{i=1}^{n} x_{i,t} \leq m,\ \forall t = 1, \ldots, T, \tag{1.9}$$

$$0 \leq y_{i,t} - y_{i,t-1} \leq t x_{i,t},\ \forall i = 1, \ldots, n,\ \forall t = 1, \ldots, T, \tag{1.10}$$

$$t x_{i,t} \leq y_{i,t} \leq t,\ \forall i = 1, \ldots, n,\ \forall t = 1, \ldots, T, \tag{1.11}$$

$$y_{i,0} = 0,\ \forall i = 1, \ldots, n, \tag{1.12}$$

$$x_{i,t} \in \{0, 1\},\ \forall i = 1, \ldots, n,\ \forall t = 1, \ldots, T, \tag{1.13}$$

$$y_{i,t} \in I\!R, \forall i = 1, \ldots, n,\ \forall t = 0, \ldots, T. \tag{1.14}$$

1.3 Deterministic setup with percentile objective

For every site i and every time moment t, we can calculate the penalty associated with the last time a sensor visited this site (see formula (3.2)). Let us pick $(1 - \alpha)$ % of *worst cases* among these $n \times T$ penalty values. Then instead of minimizing the *maximum* loss, we can minimize the the *average* loss taken over these $(1 - \alpha)$ % percent of worst-case penalty values. Despite the fact that this formulation is deterministic, we will demonstrate that it is equivalent to computing $(1 - \alpha)$ Conditional Value-at Risk (CVaR) for a set of random outcomes having equal probabilities $p_{i,t} = \frac{1}{nT}$.

To facilitate the further discussion, let us give the definition of Conditional Value-at-Risk (CVaR) [43, 45]. CVaR is closely related to a well-known quantitative risk measure referred to as Value-at-Risk (VaR). By definition, with respect to a specified probability level $(1 - \alpha)$ (in many applications the value of $(1 - \alpha)$ is set rather high, e.g. 95%), the α-VaR is the lowest amount ζ such that with probability $(1 - \alpha)$, the loss will not exceed ζ, whereas for continuous distributions the α-CVaR is the conditional expectation of losses *above* that amount ζ. As it can be seen, CVaR is a more conservative risk measure than VaR, which means that minimizing or restricting CVaR in optimization problems provides more robust solutions with respect to the risk of high losses.

Formally, α-CVaR can be expressed as

$$\text{CVaR}_\alpha(\mathbf{x}) \ = \ (1 - \alpha)^{-1} \int_{L(\mathbf{x},\mathbf{y}) \geq \zeta_\alpha(\mathbf{x})} L(\mathbf{x}, \mathbf{y})\, p(\mathbf{y})\, d\mathbf{y}, \tag{1.15}$$

where $L(\mathbf{x}, \mathbf{y})$ is a random variable driven by decision vector x and having a distribution in $I\!R$ induced by that of the vector of uncertain parameters $\mathbf{y}$.

CVaR is defined in a similar way for discrete or mixed distributions. The reader can find the formal definition of CVaR for general case in [43, 45].

It has been shown by Rockafellar and Uryasev [42, 28] that minimizing $\phi_\alpha(\mathbf{x})$ is equivalent to minimizing the function

$$F_\alpha(\mathbf{x}, \zeta) \ = \ \zeta + (1 - \alpha)^{-1} \int_{y \in R^m} [L(\mathbf{x}, \mathbf{y}) - \zeta]^+ \, p(\mathbf{y})\, d\mathbf{y}, \tag{1.16}$$

where $[t]^+ = t$ when $t > 0$ but $[t]^+ = 0$ when $t \leq 0$, and the variable ζ corresponds to the VaR value, as introduced above.

Thus we can generalize our formulation and write the objective function for our problem as

$$\min_{x,y} \text{CVaR}_\alpha \left[L(x, y, i, t)) \right], \tag{1.17}$$

where

$$L(x, y, i, t) = a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t}) \tag{1.18}$$

The particular extreme case when $\alpha \to 1$ corresponds to minimizing maximum penalty over all t-s and i-s. This case corresponds to the problem (3.32)-(3.35) formulated in section 3.2. The other extreme case $\alpha = 0$ gives average taken over all time points and sites (if we assume uniform distribution). In the latter case we care about average loss and there is a high chance of not paying enough attention to particular bad outcomes.

Generally, α indicates the *level of conservatism* the decision maker is willing to accept. The closer α approaches to 1, the narrower the range of worst cases becomes in the corresponding optimization problem.

Using the general approach outlined in formulas (3.26)-(3.27), our problem is now formulated as follows:

$$\min_{x,y,\zeta} \zeta + \frac{1}{1-\alpha} \sum_{i,t} p_{i,t} \left[a_i(1-x_{i,t}) + b_{i,t}(t-y_{i,t}) - \zeta\right]^+ \tag{1.19}$$

s.t. constraints (3.9)-(3.34),

$$\zeta \in I\!R, \tag{1.20}$$

where the values of $p_{i,t}$ can all be set equal to $1/nT$ as indicated in the beginning of this section.

Furthermore, this problem formulation can be easily transformed to a linear mixed integer problem by introducing a set of artificial variables $z_{i,t}$ that will lead to the following problem with a set of $n \times T$ additional constraints.

$$\min_{x,y,\zeta} \zeta + \frac{1}{nT(1-\alpha)} \sum_{i,t} y_{i,t} \tag{1.21}$$

$$\text{s.t. } a_i(1-x_{i,t}) + b_{i,t}(t-y_{i,t}) - \eta \le y_{i,t} \tag{1.22}$$

$$y_{i,t} \ge 0 \tag{1.23}$$

constraints (3.9)-(3.34),

$$\zeta \in I\!R, \tag{1.24}$$

1.4 Problem setup under uncertainty

To extend the proposed problem formulations to a more realistic setup, where the values of the penalty parameters are uncertain, we propose a new CVaR-based formulation of multi-sensor scheduling problems.

In this setup, assume that the fixed and variable penalty values $\mathbf{a_i}$ and $\mathbf{b_{i,t}}$ are random variables with given joint distributions. Further, we can consider a set of penalty values $(a_i^s, b_{i,t}^s)$, $s = 1, ..., S$ corresponding to S discrete samples (or *scenarios*) as an approximation of the joint distribution. Then for each $s = 1, ..., S$ the loss function can be written as:

$$L(x, y, i, t, s) = a_i^s(1-x_{i,t}) + b_{i,t}^s(t-y_{i,t}). \tag{1.25}$$

It is appropriate to consider $(1-\alpha)$ % of worst-case penalties over all indices i, t, s. We can then chose a measure of loss as an average over these $(1-\alpha)$ % worst cases and minimize the average. Namely we minimize

$$\text{CVaR}_\alpha \left[L(x, y, i, t, s)\right]. \tag{1.26}$$

Using the approach described in the previous section, we obtain the following robust optimization problem that explicitly takes into account the uncertain penalty parameters:

$$\min_{x,y,\zeta} \zeta + \frac{1}{nTS(1-\alpha)} \sum_{i,t,s} \left[\left(a_i^s(1-x_{i,t}) + b_{i,t}^s(t-y_{i,t})\right) - \zeta\right]^+ \tag{1.27}$$

s.t.

constraints (3.9)-(3.34), (2.26).

As mentioned above, this formulation can be linearized by introducing extra variables and constraints, and the linear mixed integer formulation is provided below.

$$\min_{x,y,\zeta} \zeta + \frac{1}{nTS(1-\alpha)} \sum_{i,t,s} y_{i,t,s} \tag{1.28}$$

s.t.

$$a_i^s(1-x_{i,t}) + b_{i,t}^s(t-y_{i,t}) - \zeta \le y_{i,t,s} \tag{1.29}$$

$$y_{i,t,s} \ge 0 \tag{1.30}$$

constraints (3.9)-(3.34), (2.26).

1.5 Equivalent formulations of the considered problems using cardinality constraints

In this section, we show that the developed linear mixed integer programming problems can be equivalently reformulated as problems with cardinality constraints. As it will be discussed later, solving these equivalent reformulations can provide better computational speed and performance in finding near-optimal solutions of the considered problems. It should be noted that due to the high dimensionality and complexity of these problems, it is often impossible to find exact optimal solutions in a reasonable time; however, it is often useful in practice to utilize heuristic techniques that can find near-optimal solutions fast.

There exist heuristics [18] as well as software packages [1] which can solve optimization problems formulated in terms of cardinality constraints. Cardinality function simply equals to the number of non-zero components of its vector argument. More formally, for $x = (x_1, ..., x_n)^T \in I\!R^n$

$$\text{card}(x) = \sum_{i=1}^{n} I(x_i),$$

where I is an indicator function defined as:

$$I(z) = \begin{cases} 1, & z \ne 0; \\ 0, & \text{otherwise}. \end{cases} \tag{1.31}$$

This section presents a problem formulated in terms of cardinality function. This new problem is equivalent to the initial formulation (3.32)-(3.35) that can be written as

Problem (I):

$$\min \max_{i,t} \{a_i(1-x_{i,t}) + b_{i,t}(t-y_{i,t})\}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_{i,t} \le m, \ \forall t = 1, \ldots, T, \tag{1.32}$$

$$0 \le y_{i,t} - y_{i,t-1} \le t x_{i,t}, \ \forall i = 1, \ldots, n, \ \forall t = 1, \ldots, T, \tag{1.33}$$

$$t x_{i,t} \le y_{i,t} \le t, \ \forall i = 1, \ldots, n, \ \forall t = 1, \ldots, T, \tag{1.34}$$

$$y_{i,0} = 0, \ \forall i = 1, \ldots, n, \tag{1.35}$$

$$x_{i,t} \in \{0, 1\}, \ \forall i = 1, \ldots, n, \ \forall t = 1, \ldots, T, \tag{1.36}$$

$$y_{i,t} \in I\!R, \forall i = 1, \ldots, n, \ \forall t = 0, \ldots, T. \tag{1.37}$$

The new problem can be formulated as:

Problem (II):

$$\begin{aligned}
& \min \max_{i,t} \{a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})\} && \\
\text{s.t.} \quad & \text{card}(\bar{x}_t) \le m, \ \forall t = 1, \dots, T, && (1.38) \\
& 0 \le y_{i,t} - y_{i,t-1} \le t x_{i,t}, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, && (1.39) \\
& y_{i,t} \le t, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, && (1.40) \\
& y_{i,0} = 0, \ \forall i = 1, \dots, n, && (1.41) \\
& 0 \le x_{i,t} \le 1, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, && (1.42) \\
& y_{i,t} \in I\!R, \forall i = 1, \dots, n, \ \forall t = 0, \dots, T. && (1.43)
\end{aligned}$$

where $\bar{x}_t = (x_{1,t}, ..., x_{N,t})^T$.

The following theorem provides the relation between the two problems

Theorem 1. *The set of optimal solutions of problem (I) belongs to the set of optimal solutions of problem (II). Moreover, if a point (x^{II}, y^{II}) is an optimal solution for (II) then the optimal solution of (I) (x^I, y^I) can be constructed as*

$$\begin{aligned}
x^I_{i,t} &= \lceil x^{II}_{i,t} \rceil, \\
y^I_{i,t} &= \max_{\tau \le t} \{\tau x^I_{i,\tau}\}.
\end{aligned}$$

In order to prove the theorem we will use an auxiliary formulation.

Problem (III):

$$\begin{aligned}
& \min \max_{i,t} \{a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})\} \\
\text{s.t.} \quad & \text{card}(\bar{x}_t) \le m, \ \forall t = 1, \dots, T, \\
& 0 \le y_{i,t} - y_{i,t-1} \le t x_{i,t}, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, \\
& y_{i,t} \le t, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, \\
& y_{i,0} = 0, \ \forall i = 1, \dots, n, \\
& 0 \le x_{i,t} \le 1, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, \\
& y_{i,t} \in I\!R, \forall i = 1, \dots, n, \ \forall t = 0, \dots, T.
\end{aligned}$$

Denote by $z^{(I)}$, $z^{(II)}$ and $z^{(III)}$ the optimal objective values of problems (I)-(III) consequently.

Lemma 1. *For every optimal solution of (III) there exists a solution that will be both feasible and optimal in (I) and (III) (i.e. formulations (I) and (III) are equivalent in this sense).*

Proof. Equations (1.33)-(1.34) enforce that

$$y_{i,t} = \max_{\tau \le t} \{\tau x_{i,\tau}\}.$$

If there exists an optimal solution for (III) such that $x_{i,t} = 1$ but $y_{i,t} < t$ for some i, t (i.e. it is not feasible in (I)) then you can build a new solution that has the same values of $x^*_{i,t} = x_{i,t} \ \forall i, t$ and $y^*_{i,t} = \max_{\tau \le t}\{\tau x_{i,\tau}\}$. This solution will be feasible for both formulations (I) and (III).

Moreover, $\forall i, t \ y_{i,t} <= y^*_{i,t}$, i.e. the objective value will not increase and, therefore, this solution will be also optimal for (III). Obviously this solution will be feasible and optimal for formulation (I) (since $z^{(III)} \le z^{(I)}$).

Thus, for every optimal solution of (III) there exists a solution that will be both feasible and optimal in (I) and (III) that means that these two formulations are equivalent. □ □

Proof of Theorem 1. Let us consider some optimal solution of (II) $x^0_{i,t}$, $y^0_{i,t}$ and build a new solution $x^*_{i,t} = I\{x^0_{i,t} > 0\}$; $y^*_{i,t} = y^0_{i,t}$. This solution will still be feasible and optimal for (II) since it will not increase the objective value.

Obviously, this solution will be feasible and optimal for (III) ($z^{(II)} \leq z^{(III)}$).

According to Lemma 1 for every optimal solution of (III) there exists a solution $x^{**}_{i,t} = x^*_{i,t}, y^{**}_{i,t} = \max_{\tau \leq t}\{\tau x^*_{i,\tau}\}$ that will be both feasible and optimal for (I) and (III).

This solution will be integer and feasible for (II). Since $\forall\ i,t\ y^*_{i,t} \leq y^{**}_{i,t}$ then the objective value will not increase and, therefore, this solution will also be optimal for (II).

Thus, there always exists the optimal integer solution for (II) that will be also optimal for (I) □

In order to prevent the solution of (II) from being non-integral, we add a penalty to the objective of (II):

Problem (IV):

$$\begin{aligned} &\min \max_{i,t}\{a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})\} + \lambda \cdot (m \cdot T - \sum_{i,t} x_{i,t}) \\ \text{s.t.}\quad &\text{constraints (1.38)-(1.43)}, \end{aligned}$$

where $\lambda > 0$.

Corollary 1. *Problems (I) and (IV) have the same set of optimal values of* $\{x_{i,t}\}$.

Similar theorems can be proven for the other formulations, namely percentile deterministic and stochastic setups. For deterministic case problem (Ia), which is equivalent to (2.25) is related to reformulated in terms of cardinality problem (IIa):

Problem (Ia):

$$\begin{aligned} &\min \mathrm{CVaR}_\alpha\{a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})\} \\ \text{s.t.}\quad &\text{constraints (1.32)-(1.37).} \end{aligned}$$

Problem (IIa):

$$\begin{aligned} &\min \mathrm{CVaR}_\alpha\{a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})\} \\ \text{s.t.}\quad &\text{constraints (1.38)-(1.43).} \end{aligned}$$

Theorem 2. *The set of optimal solutions of problem (Ia) belongs to the set of optimal solutions of problem (IIa). Moreover if a point (x^{II}, y^{II}) is an optimal solution for (IIa) then the optimal solution of (Ia) (x^I, y^I) can be constructed as*

$$\begin{aligned} x^I_{i,t} &= \lceil x^{II}_{i,t} \rceil, \\ y^I_{i,t} &= \max_{\tau \leq t}\{\tau x^I_{i,\tau}\}. \end{aligned}$$

For the stochastic case problem (Ib), which is equivalent to (2.27) is related to the reformulated in terms of cardinality problem (IIb):

Problem (Ib):

$$\begin{aligned} &\min \mathrm{CVaR}_\alpha\{a^s_i(1 - x_{i,t}) + b^s_{i,t}(t - y_{i,t})\} \\ \text{s.t.}\quad &\text{constraints (1.32)-(1.37).} \end{aligned}$$

Problem (IIb):

$$\begin{aligned} &\min \mathrm{CVaR}_\alpha\{a^s_i(1 - x_{i,t}) + b^s_{i,t}(t - y_{i,t})\} \\ \text{s.t.}\quad &\text{constraints (1.38)-(1.43).} \end{aligned}$$

Theorem 3. *The set of optimal solutions of problem (Ib) belongs to the set of optimal solutions of problem (IIb). Moreover if a point (x^{II}, y^{II}) is an optimal solution for (IIb) then the optimal solution of (Ib) (x^I, y^I) can be constructed as*

$$x^I_{i,t} = \lceil x^{II}_{i,t} \rceil,$$
$$y^I_{i,t} = \max_{\tau \le t}\{\tau x^I_{i,\tau}\}.$$

1.6 Sensor Scheduling in Network-based Settings

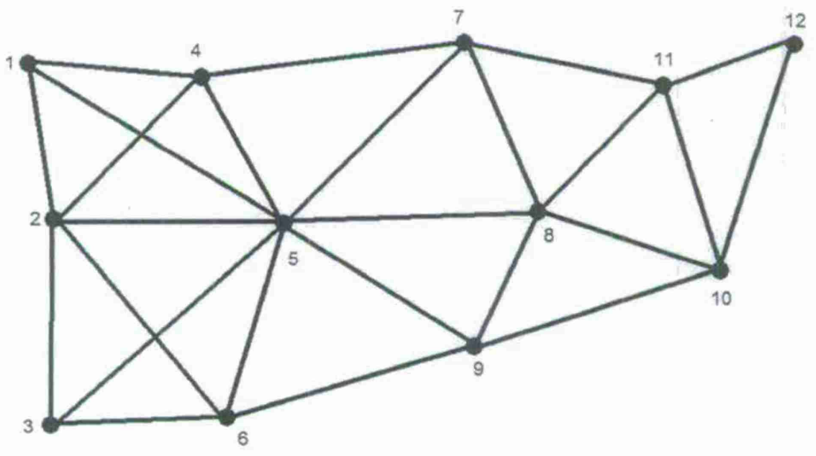

Figure 1.1: Example of a possible network. Two nodes are connected by an arc if a sensor can move from one node to another in consequent time periods.

First, let us discuss a special case of one sensor ($m = 1$) to give an idea of this modeling approach. In case when surveillance requires sensors to physically move from one site to another their transition abilities are limited with distance or other constraint (for example a mountain can be a natural obstacle for UAV to move between sites). In this case each site can be modeled as a node of a network $G = (V, E)$. Whenever there is no arc between two nodes i and j we add the inequality

$$x_{i,t} + x_{j,t+1} \le 1, \tag{1.44}$$

that prohibits the infeasible move $i \rightarrow j$ in consequent time periods t and $t + 1$. Formulations (3.32)-(3.35), (2.25) and (2.27) can be slightly modified to obtain corresponding formulations for one sensor ($m = 1$). If we demand the sensor to start and come back to a depot located at the certain site we can optionally set the initial ($i_0 \in \{1, ..., n\}$) and final ($i_T \in \{1, ..., n\}$) locations of the sensor.

Thus for the special case when $m = 1$ problem (3.32)-(3.35) can be formulated on the network:

$$\min \max_{i,t}\{a_i(1 - z_{i,t}) + b_{i,t}(t - y_{i,t})\} \tag{1.45}$$

s.t. constraints (3.9)-(3.34), ($m = 1$),

$$x_{i,t} + x_{j,t+1} \le 1 \text{ whenever } (i,j) \notin E, \forall t = 1, \dots T,\ i,j = 1, \dots, n \tag{1.46}$$

$$x_{i_0,1} = 1, \text{ where } i_0 \in \{1, \dots, n\} \text{ is the initial location of the sensor} \tag{1.47}$$

$$x_{i_T,1} = 1, \text{ where } i_T \in \{1, \dots, n\} \text{ is the final location of the sensor} \tag{1.48}$$

In this formulation constraints (1.46) prohibit infeasible moves between not connected nodes. (1.47) and (1.48) set initial and final destination for the sensor. The other formulations, namely deterministic (2.25) and stochastic (2.27), can be easily adapted for network case ($m = 1$) in the same way by adding network constraints (1.46)-(1.48) to the existing sets of constraints.

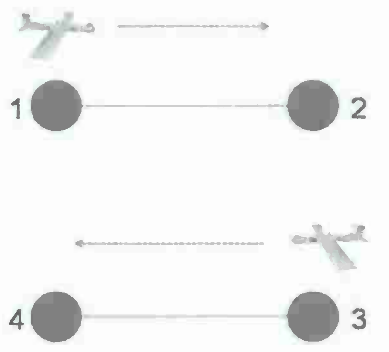

Figure 1.2: Counterexample ($m = 2$): two sensors cannot perform simultaneous feasible move due to the constraint $x_{1,1} + x_{4,2} \leq 1$

This approach, however, may not be easily extended on cases of two or more sensors. If we simply add (1.46)-(1.48) to existing non-network formulations we can arrive at the situation which prevents feasible moves when two or more sensors are involved. Figure 1.2 provides a counterexample. Let two sensors at time moment $t = 1$ are located at nodes 1 and 3. Though they could move to nodes 2 and 4 respectively at the next time point $t = 2$ the constraint $x_{1,1} + x_{2,4} \leq 1$ would prohibit this move. To avoid such a situation we need to add one more index for decision variable x:

$$x_{i,t,k} = \begin{cases} 1, & \text{sensor } k \text{ is surveiling site } i \text{ at time } t; \\ 0, & \text{otherwise.} \end{cases} \tag{1.49}$$

We can ensure that every sensor is assigned to a site at every time period T with the constraint:

$$\sum_{i=1}^{n} x_{i,t,k} = 1, \; \forall k = 1, \ldots, m, \; \forall t = 1, \ldots, T. \tag{1.50}$$

Let us introduce $z_{i,t}$ indicating whether site i is observed at time t, namely

$$z_{i,t} = \begin{cases} 1, & \text{if any sensor is surveiling site } i \text{ at time } t; \\ 0, & \text{otherwise.} \end{cases} \tag{1.51}$$

Variables $z_{i,t}$ and $x_{i,t,k}$ can be related with the constraint

$$z_{i,t} \leq \sum_{i=1}^{n} x_{i,t,k} \leq m \cdot z_{i,t}, \tag{1.52}$$

which states that site i is being observed at time t ($z_{i,t} = 1$) if and only if at least one sensor is present at site i ($\sum_{i=1}^{n} x_{i,t,k} > 0$).

The loss function is written as

$$L(x, y, i, t) = a_i^s(1 - z_{i,t}) + b_{i,t}^s(t - y_{i,t}). \tag{1.53}$$

If we want to minimize maximum loss then using the formulated above constraints and based on the problem formulated for non-network setup we can reformulate the deterministic maximum loss minimization problem (3.32)-(3.35) on network as follows:

$$\min \max_{i,t} \{a_i(1 - z_{i,t}) + b_{i,t}(t - y_{i,t})\} \tag{1.54}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_{i,t,k} = 1, \ \forall k = 1, \dots, m, \ \forall t = 1, \dots, T, \tag{1.55}$$

$$z_{i,t} \le \sum_{k=1}^{m} x_{i,t,k} \le m \cdot z_{i,t}, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, \tag{1.56}$$

$$0 \le y_{i,t} - y_{i,t-1} \le t z_{i,t}, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, \tag{1.57}$$

$$t z_{i,t} \le y_{i,t} \le t, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, \tag{1.58}$$

$$y_{i,0} = 0, \ \forall i = 1, \dots, n, \tag{1.59}$$

$$x_{i,t,k} + x_{j,t+1,k} \le 1, \quad \text{whenever } (i,j) \notin E, \forall t = 1, \dots T, \ i,j = 1, \dots, n, \ k = 1, \dots, m \tag{1.60}$$

$$x_{i_{0,k},1,k} = 1, \quad \text{where } i_{0,k} \in \{1, \dots, n\} \text{ is the initial location of sensor } k \tag{1.61}$$

$$x_{i_{T,k},T,k} = 1, \quad \text{where } i_{T,k} \in \{1, \dots, n\} \text{ is the final location of sensor } k, \tag{1.62}$$

$$x_{i,t,k} \in \{0,1\}, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, \ \forall k = 1, \dots, m, \tag{1.63}$$

$$y_{i,t} \in \mathbb{R}, \forall i = 1, \dots, n, \ \forall t = 0, \dots, T, \tag{1.64}$$

$$z_{i,t} \in \{0,1\}, \forall i = 1, \dots, n, \ \forall t = 1, \dots, T. \tag{1.65}$$

Formulations for CVaR stochastic and deterministic cases as well as the linearized formulation can be obtained the same way as in non network case.

1.7 Computational Experiments

The computational experiments were performed on the test problems using two commercial optimization software solvers: ILOG CPLEX [2] and AOrDA PSG [1]. The performance of the solvers is compared in tables 1.1-1.3 (each table corresponds to one of the three problem formulations). It can be observed that CPLEX finds exact solutions, however, it takes too much time for large instances, especially for the problems under uncertainty. PSG allows sacrificing quality for time, i.e. the obtained solutions for cardinality formulations are not globally optimal, but the computational time is negligibly small. The numerical experiments shows that the local solutions differ from global in 10-20 % for most cases.

Table 1.4 compares performance of CPLEX and PSG in finding approximate solutions for stochastic case ($n = 12$ sites and $T = 10$ time periods). We stopped CPLEX when it found the objective as small as the PSG objective value (and recorded the computation time). It appears that PSG outperforms CPLEX for problems with large number of stochastic scenarios while they have similar performance for small size problems. Therefore, based on the size of the problem and user requirements, one can determine the appropriate equivalent problem formulation and the optimization solver that can be used to find an optimal or a near-optimal solution.

We performed experiments on network formulation using the network provided on figure 1.1. Table 1.5 provides CPU times in seconds for obtaining exact solution for the deterministic network case (1.54)-(1.65) in ILOG CPLEX. Computation was performed for number of sites from 6 to 12 and 10 discrete time steps.

		n=8	n=9	n=10	n=11	n=12
m=1	cplex value	320	330	330	332	332
	psg value	375	376	376	370	375
	%	14.7%	12.2%	12.2%	10.3%	11.5%
	time: cplex/psg	31.2/2.2	79.1/2.4	134.9/2.5	167.7/2.7	198.9/2.7
m=2	cplex value	240	245	250	260	265
	psg value	305	310	304	310	310
	%	21.3%	21%	17.8%	16.1%	14.5%
	time: cplex/psg	38.4/2.2	142.7/2.3	928/2.5	2042.7/2.6	5898.9/2.8
m=3	cplex value	206	210	215	217	224
	psg value	233	250	265	256	275
	%	11.6%	16%	18.9%	15.2%	18.5%
	time: cplex/psg	30.7/2.3	39.3/2.4	81.1/2.5	1003.6/2.7	9317.9/2.9
m=4	cplex value	190	194	196	200	200
	psg value	215	217	242	237	242
	%	11.6%	10.6%	19%	15.6%	17.4%
	time: cplex/psg	6/2.3	76.5/2.5	64.6/2.6	231.4/2.7	589.3/2.8
m=5	cplex value	183	185	188	190	190
	psg value	196	202	215	217	220
	%	6.6%	8.4%	12.6%	12.4%	13.6%
	time: cplex/psg	1.4/2.3	2.2/2.5	38/2.6	51.9/2.7	68/3
m=6	cplex value	165	170	170	183	185
	psg value	185	185	197	195	200
	%	10.8%	8.1%	13.7%	6.2%	7.5%
	time: cplex/psg	1.1/2.4	1.5/2.5	2.9/2.7	29.1/2.8	123/3
m=7	cplex value	155	160	163	168	170
	psg value	160	171	185	190	188
	%	3.1%	6.4%	11.9%	11.6%	9.6%
	time: cplex/psg	0.3/2.3	0.9/2.5	1.8/2.8	113.2/2.9	25.2/3

Table 1.1: Performance results for deterministic model (3.32)-(3.35). n - number of sites; m - number of sensors. The number of discrete time steps is fixed: $T = 10$.

		n=8	n=9	n=10	n=11	n=12
m=1	cplex value	307.1	318.4	317	318.7	318
	psg value	360.1	357.9	356.8	357.6	353.7
	%	14.7%	11%	11.2%	10.9%	10.1%
	time: cplex/psg	45.5/2.8	99.3/3	74/2.9	62.8/3.1	130.9/3
m=2	cplex value	231.9	241.1	245.2	-	-
	psg value	297	299	297.5	293.5	291.6
	%	21.9%	19.4%	17.6%	-	-
	time: cplex/psg	1072.8/3	10924.6/3.1	16212.1/3.1	-/3.2	-/3.2
m=3	cplex value	198.6	205.3	-	-	-
	psg value	223.6	237.3	245	255.4	260.9
	%	11.2%	13.5%	-	-	-
	time: cplex/psg	1910.7/2.8	45540.4/2.8	-/2.9	-/3	-/3.4
m=4	cplex value	187.9	190.4	-	-	-
	psg value	211.4	207.2	230	218.2	221.8
	%	11.1%	8.1%	-	-	-
	time: cplex/psg	7989.2/3.2	23852.7/3	-/3.3	-/3.4	-/3.2
m=5	cplex value	173.3	179.3	-	-	-
	psg value	193.4	192.6	197.8	207.7	210
	%	10.4%	6.9%	-	-	-
	time: cplex/psg	7672.6/2.9	25593/2.8	-/3.3	-/3.3	-/3.7
m=6	cplex value	161.1	165.3	-	-	-
	psg value	179.5	182.8	185.2	195.2	190.9
	%	10.2%	9.5%	-	-	-
	time: cplex/psg	2250.3/2.8	36618.1/3.3	-/3.4	-/2.9	-/3.4
m=7	cplex value	142.4	-	-	-	-
	psg value	150.9	166.6	173.4	179	183.8
	%	5.6%	-	-	-	-
	time: cplex/psg	6640.9/2.9	65122.8/3.3	-/3	-/3	-/3.7

Table 1.2: Performance results for CVaR type deterministic model (2.25). n - number of sites; m - number of sensors. The number of discrete time steps is fixed: $T = 10$. CVaR confidence level $\alpha = 0.9$.

		n=8	n=9	n=10	n=11	n=12
m=1	cplex value	-	-	-	-	-
	psg value	394	398.7	390.1	395.6	394.9
	%	-	-	-	-	-
	time: cplex/psg	-/25.9	-/28	-/35.8	-/40.2	-/47.7
m=2	cplex value	-	-	-	-	-
	psg value	297.9	304	321.7	324.4	318.6
	%	-	-	-	-	-
	time: cplex/psg	-/32.7	-/38.6	-/51.4	-/67.8	-/76
m=3	cplex value	-	-	-	-	-
	psg value	260.4	260.1	262.4	264.4	269
	%	-	-	-	-	-
	time: cplex/psg	-/38.4	-/45.4	-/56.3	-/71.4	-/84.6
m=4	cplex value	-	-	-	-	-
	psg value	229.4	231.6	242.9	236.3	250.8
	%	-	-	-	-	-
	time: cplex/psg	-/47.2	-/60.2	-/72.5	-/84.6	-/98.4
m=5	cplex value	-	-	-	-	-
	psg value	209.1	212.4	220.8	224.6	230.1
	%	-	-	-	-	-
	time: cplex/psg	-/56.1	-/72.3	-/83.7	-/96.6	-/119.5
m=6	cplex value	-	-	-	-	-
	psg value	193.2	199.9	209.2	210.3	218.7
	%	-	-	-	-	-
	time: cplex/psg	-/51.8	-/67.3	-/86.1	-/112.9	-/121.4
m=7	cplex value	-	-	-	-	-
	psg value	164.4	186.3	187.6	198.6	204.4
	%	-	-	-	-	-
	time: cplex/psg	-/45	-/66.5	-/89.8	-/111.3	-/133.5

Table 1.3: Performance results for CVaR type stochastic model (2.27). n - number of sites; m - number of sensors. The number of discrete time steps is fixed: $T = 10$, number of scenarios $S = 100$. CVaR confidence level $\alpha = 0.9$.

	PSG Value	PSG Time (sec)	CPLEX Time (sec)
m=1	389.2	103	32
m=2	318.532	110	185
m=3	276.332	133	230
m=4	246.648	136	247
m=5	229.234	208	210
m=6	218.166	158	330
m=7	204.1	201	260
m=8	193.199	191	220
m=9	180.976	155	250
m=10	164.746	191	280
m=11	146.607	180	200

Table 1.4: Comparing PSG and CPLEX performance for obtaining approximate solution of CVaR type stochastic problem (2.27)($n = 12$ sites and $T = 10$ time periods).

	n=6	n=7	n=8	n=9	n=10	n=11	n=12
m=1	0.31	0.77	0.76	2.16	0.90	0.50	1.21
m=2	3.96	3.96	8.00	15.85	238.42	174.93	315.76
m=3	0.93	21.48	22.20	9.17	340.86	350.85	2037.73
m=4	0.20	0.31	0.74	34.26	14.45	926.64	436.80
m=5	0.32	1.79	2.62	31.40	91.38	62.20	3359.69
m=6		0.58	1.79	4.36	69.76	19.05	238.59
m=7			2.12	4.08	22.87	19.57	47.20
m=8				0.79	6.29	7.14	38.69
m=9					1.17	3.19	6.81
m=10						0.79	2.15
m=11							1.94

Table 1.5: ILOG CPLEX CPU time (sec) for network deterministic model (1.54)-(1.65). n - number of sites; m - number of sensors. The number of discrete time steps is $T = 10$.

1.8 Conclusion

The paper develops a mathematical programming techniques for solving a class of multi-sensor scheduling problems. Three robust optimization problems have been formulated: two for deterministic and one for stochastic case. The obtained 0-1 problems have also been reformulated in terms of cardinality functions.

Numerical experiments are conducted using two commercial solvers ILOG CPLEX and AOrDa PSG. CPLEX gives exact solutions for small problems. Both solvers give an approximate solution in reasonable time for large problems. However, for large stochastic problems with many scenarios, solving the reformulated problems with cardinality constraints using PSG provided good quality approximate solutions faster than CPLEX. Therefore, this approach can be beneficial in the settings of time-critical systems where computational speed of finding good approximate solutions is the crucial factor.

Chapter 2

Mathematical Programming Techniques for Sensor Networks

This chapter presents the results published in Alexey Sorokin, Nikita Boyko, Nikita Boyko , Stan Uryasev and Panos M Pardalos. Panos M Pardalos. Algorithms, pp. 565-581, 2009.

2.1 Introduction

The area of sensor networks research has recently gained a great deal of attention [12, 37, 16, 3, 4, 36, 17, 20, 33, 11]. Various sensors are used in both civilian and military tasks. Sensing devices can be deployed in either static or dynamic settings, where the positions of each sensor can be permanent or dynamically changing (such as in the case of sensors installed on air vehicles). Multiple sensor systems are commonly represented as networks, since besides collecting important information, sensing devices can transmit and exchange information via wireless communication between sensor nodes. Therefore, network (graph) structures are convenient and informative in terms of efficient representation of the structure and properties thereof. To analyze and optimize the performance of sensor networks, mathematical programming techniques are extensively used.

The purpose of this paper is to give a brief review of some of the recent developments in mathematical programming as applied to sensor network research. Various types of optimization problems can be formulated and solved in this context [7, 29, 33]. Moreover, the pursued tasks can vary from optimizing the network performance to network interdiction, where the goal is to disrupt enemy networks by interfering with communication network integrity. We will outline the formulations and briefly describe the solution methods used to tackle these problems.

In many cases, the identified optimization problems are challenging from the computational viewpoint. In particular, the theoretical computational complexity of many of these problems was proven to be *NP-hard*. Therefore, efficient algorithms need to be developed to ensure that the near-optimal solutions are found quickly. This is essential to ensure that the decisions regarding efficient operations of sensor networks could be made in a real-time mode, which can be crucial in many applications. Moreover, the uncertain factors that commonly arise in real-world situations also need to be incorporated in the mathematical programming problems, which makes them even more challenging to formulate and solve. In this chapter, we will address some of the challenges arising in this area.

In particular, we will describe several important classes of problems that have recently been addressed in the literature. We will start the discussion with the description of recent promising developments in the area of sensor network localization, which allow identifying global positions of all the nodes in a network using limited and sometimes noisy information. It turns out that semidefinite programming techniques can be efficiently used to tackle these problems. Next, we will discuss the problems of single and multiple sensor scheduling for area surveillance, including the setups under uncertainty. We will also address the issues of wireless communication network connectivity and integrity, as well as network interdiction problems.

2.2 Sensor Network Localization

The wide range of sensors applications reveal different requirements to the network topology identification [32, 30, 15, 24]. For example network parameters can be influenced by land surface, transmission characteristics, energy consumption policy, etc. Ad hoc and dynamic networks also require identification of node coordinates. Such problems are intensively studied in the literature. Utilizing mathematical programming techniques often allows to find efficient solutions.

2.2.1 Positioning Using Angle of Arrival

In many cases, practical situations require one to be aware of a sensor's physical coordinates. Installing GPS receivers in every sensor is not always optimal from the cost-related and other perspectives. Typically, only a few nodes (seed nodes, landmarks, etc) of the network are equipped with GPS and know their physical location. The rest of the nodes can only communicate with other nodes and determine relative location characteristics such as distance, angles, etc. Various localization techniques are used to obtain location of all the sensors in the network. The following method assumes that the network consists of two types of nodes: usual and more capable nodes - which knows its position. Niculescu and Nath propose a method by which nodes in an ad hoc network collaborate in finding their position and orientation, assuming that a small part of the network has a position capability. Also, every node in the network has a capability to determine the angle of the arriving signal (AOA).

Each node in the network has one fixed main axis (which may not be the same for different nodes) and the node is able to measure all angles against this axis (Figure 2.1).

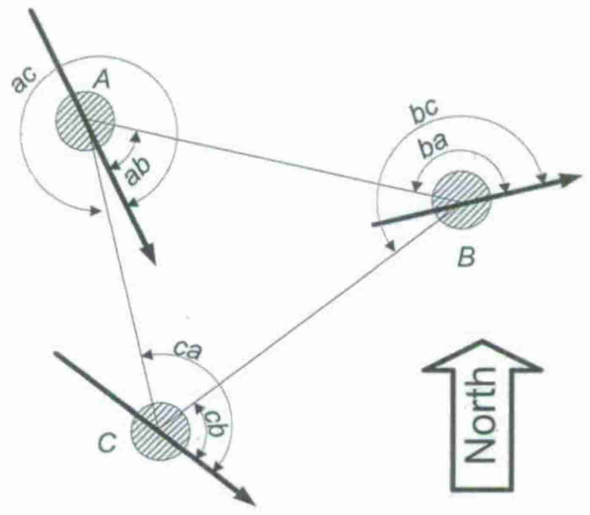

Figure 2.1: Nodes with AOA capability.

Every node in an ad hoc network can only communicate with its immediate neighbors within the radio range, and its neighbors may not always be landmarks, i.e., the nodes that know their position. Niculescu Nath propose in [19] a method to forward orientation in such way, that the nodes which are not in direct contact with landmarks can determine its orientation with respect to the landmarks. Orientation means bearing, radial, or both. Bearing is an angle measurement with respect to another object. A radial is the angle under which the object is seen from another point. The authors examine two algorithms: Distance Vector Bearing (DV-Bearing), which allows each node to get a bearing to landmark, and DV-Radial, which allows a node to get a bearing and radial to a landmark. The propagation in both algorithms works the following way: nodes adjacent to landmarks determine their bearing/radial directly from landmark and send to the network the information about their position. The method of computing node's bearing and radial at each step is shown in Figure 2.2.

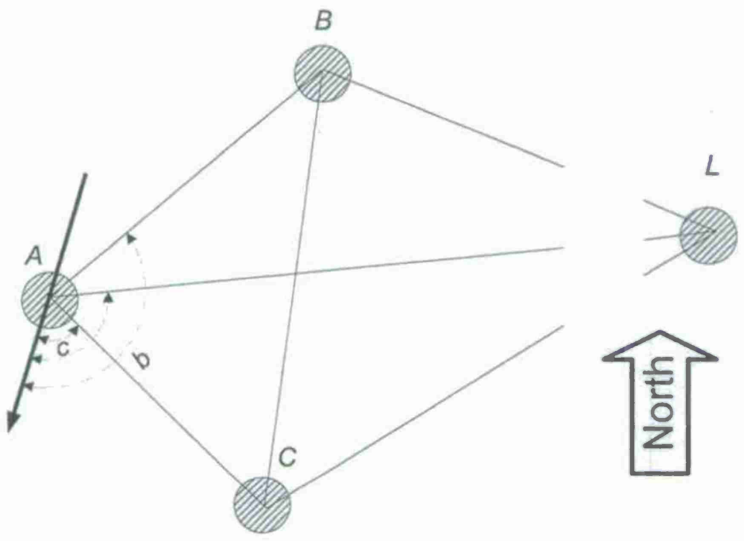

Figure 2.2: Node A computes its bearing to L using information from B and C

DV-Bearing algorithm works the following way. Nodes A, B, and C are neighbors and they can communicate with each other. Suppose that the node A needs to find its bearing to node L, which is not within radio range of node A but within radio range of nodes B and C. Since A, B, and C can locate each other than the node A can determine all the angles in triangles ABC and BCL. But that would allow to calculate the angle LAC and consequently the bearing of A to L, which is equal to $c + LAC$. Once node A knows three bearings to landmarks, which are not at the same line, then it can calculate its own location by triangulation.

The DV-Radial algorithm works the same way but with only one difference that node A needs to know not only bearings of nodes B and C to node L, but also the radials of B and C from L. The knowledge of radials improves accuracy of the algorithm. When all angles are measured against the same direction (for example, when compass is available) then these two methods become identical.

2.2.2 Semidefinite Programming (SDP) for Sensor Network Localization

The section considers localization problem when information on distances for anchor nodes and unknown sensors nodes are given. Suppose we consider localization problem on the plane. We have m known points $a_k \in \mathbb{R}^2$, $k = 1, ..., m$ and n unknown nodes $x_j \in \mathbb{R}^2$, $j = 1, ..., n$. Let us consider three sets of node pairs N_e, N_l, N_u. For pairs in N_e we know exact distances d_{kj} between a_k and x_j and $\hat{d_{ij}}$ between x_i and x_j. N_l is a set of pairs with known lower bounds $\underline{r}_{kj}$ and $\underline{r}_{ij}$. Finally, N_u is a set of upper bounds $\overline{r}_{kj}$ and $\overline{r}_{ij}$. Naturally our goal is to minimize estimation error which immediately lead us to the following non-convex optimization problem

$$\begin{aligned}\min \sum_{(i,j)\in N_e,\, i<j} & \left| \|x_i - x_j\|^2 - \hat{d}_{ij}^2 \right| \\ + \sum_{(k,j)\in N_e} & \left| \|a_k - x_j\|^2 - d_{kj}^2 \right| \\ + \sum_{(i,j)\in N_l,\, i<j} & (\|x_i - x_j\|^2 - \underline{r}_{ij}^2)_- \\ + \sum_{(k,j)\in N_l} & (\|a_k - x_j\|^2 - \underline{r}_{kj}^2)_- \\ + \sum_{(i,j)\in N_u,\, i<j} & (\|x_i - x_j\|^2 - \overline{r}_{ij}^2)_+ \\ + \sum_{(k,j)\in N_u} & (\|a_k - x_j\|^2 - \overline{r}_{kj}^2)_+,\end{aligned} \tag{2.1}$$

where $(u)_-$ and $(u)_+$ are defined as

$$\begin{aligned}(u)_- &= \max\{0, -u\} \\ (u)_+ &= \max\{0, u\}.\end{aligned}$$

The norm of vector x is defined as $\|x\| = \sqrt{x^T x}$. In [5, 27] the authors study semidefinite relaxations of the problem.

The formulated problem can be rewritten by introducing matrix notation and slack variables as

$$\begin{aligned}&\min \sum_{(i,j)\in N_e,\, i<j} (\alpha_{ij}^+ + \alpha_{ij}^-) + \sum_{(k,j)\in N_e,} (\alpha_{kj}^+ + \alpha_{kj}^-) \\ &\qquad + \sum_{(i,j)\in N_l,\, i<j} \beta_{ij}^- + \sum_{(k,j)\in N_l} \beta_{kj}^- \\ &\qquad + \sum_{(i,j)\in N_u,\, i<j} \beta_{ij}^+ + \sum_{(k,j)\in N_u} \beta_{ij}^+ \\ &\text{s.t.} \\ &s_{ij}^T Y e_{ij} - d_{ij}2 = \alpha_{ij}^+ - \alpha_{ij}^-,\ \forall i,j \in N_e,\ i<j, \\ &(a_k; e_j)^T \begin{pmatrix} I & X \\ X^T & Y \end{pmatrix} (a_k; e_j) - (\hat{d}_{kj})^2 = \alpha_{kj}^+ - \alpha_{kj}^-, \\ &\forall k,j \in N_e, \\ &e_{ij}^T Y e_{ij} - (d_{ij})^2 \geq -\beta_{ij}^-,\ \forall i,j \in N_l,\ i<j \\ &(a_k; e_j)^T \begin{pmatrix} I & X \\ X^T & Y \end{pmatrix} (a_k; e_j) - (\underline{r}_{kj})^2 \geq -\beta_{kj}^-, \\ &\forall k,j \in N_l, \\ &e_{ij}^T Y e_{ij} - (d_{ij})^2 \leq \beta_{ij}^-,\ \forall i,j \in N_l,\ i<j \\ &(a_k; e_j)^T \begin{pmatrix} I & X \\ X^T & Y \end{pmatrix} (a_k; e_j) - (\underline{r}_{kj})^2 \leq \beta_{kj}^-, \\ &\forall k,j \in N_l, \\ &\alpha_{ij}^+, \alpha_{ij}^-, \alpha_{kj}^+, \alpha_{kj}^-, \beta_{ij}^+, \beta_{ij}^-, \beta_{kj}^+, \beta_{kj}^- \leq 0, \\ &Y = X^T X.\end{aligned}$$

Here $X = [x_1, x_2, \ldots, x_n]$ is a $2 \times n$ matrix. In order to cope with nonconvexity equality, $Y = X^T X$ is replaced with

$Y \succeq X^T X$ or equivalently

$$Z := \begin{pmatrix} I & X \\ X^T & I \end{pmatrix} \succeq 0.$$

The last relation leads to a standard SDP formulation:

$$\begin{aligned}
\min \sum_{(i,j)\in N_e,\ i<j} (\alpha_{ij}^+ + \alpha_{ij}^-) &+ \sum_{(k,j)\in N_e,} (\alpha_{kj}^+ + \alpha_{kj}^-) \\
&+ \sum_{(i,j)\in N_l,\ i<j} \beta_{ij}^- + \sum_{(k,j)\in N_l} \beta_{kj}^- \\
&+ \sum_{(i,j)\in N_u,\ i<j} \beta_{ij}^+ + \sum_{(k,j)\in N_u} \beta_{ij}^+
\end{aligned}$$

s.t.

$$\begin{aligned}
&(1;0;\mathbf{0})^T Z(1;0;\mathbf{0}) = 1, \\
&(0;1;\mathbf{0})^T Z(0;1;\mathbf{0}) = 1, \\
&(1;1;\mathbf{0})^T Z(1;1;\mathbf{0}) = 2, \\
&(\mathbf{0};e_{ij})^T Z(\mathbf{0};e_{ij}) - \alpha_{ij}^+ + \alpha_{ij}^- = (\hat{d}_{ij})^2, \forall i,j \in N_e, i<j, \\
&(a_k;e_j)^T Z(a_k;e_j) - \alpha_{kj}^+ + \alpha_{kj}^- = (\hat{d}_{kj})^2, \forall k,j \in N_e, \\
&(\mathbf{0};e_{ij})^T Z(\mathbf{0};e_{ij}) + \beta -_{ij} \geq (\underline{r}_{ij})^2, \forall i,j \in N_l, i<j, \\
&(a_k;e_j)^T Z(a_k\ e_j) + \beta_{kj}^- \geq (\underline{r}_{kj})^2, \forall k,j \in N_l, \\
&(\mathbf{0};e_{ij})^T Z(\mathbf{0};e_{ij}) - \beta +_{ij} \leq (\overline{r}_{ij})^2, \forall i,j \in N_u, i<j, \\
&(a_k;e_j)^T Z(a_k\ e_j) - \beta_{kj}^+ \leq (\overline{r}_{kj})^2, \forall k,j \in N_u, \\
&\alpha_{ij}^+, \alpha_{ij}^-, \alpha_{kj}^+, \alpha_{kj}^-, \beta_{ij}^+, \beta_{ij}^-, \beta_{kj}^+, \beta_{kj}^- \leq 0, \\
&Z \succeq 0.
\end{aligned}$$

Papers [5, 27] provide a criterion of solution existence and uniqueness, as well as statistical interpretation of the formulation in case when distance values are random values with normally distributed measurement errors. The SDP problems are solved using interior point algorithms. The numerical experiments results demonstrate the efficiency of the proposed approach.

2.3 Sensor Scheduling

Surveillance is an important task that can be effectively performed by an intelligent network of sensors. For example, satellites can be equipped with cameras to monitor Earth surface for different events, such as forest fire, border crossing or enemy hostile activity. Another example is traffic monitoring at the roads and intersections. Many scientific publications have recently appeared in the literature due to increasing interest to the problem of finding optimal schedule for sensors [21, 8, 13, 31, 26, 14]. Most common technological and budget constraint is a low number of sensors to monitor all the objects of interest simultaneously. Thus, finding the schedule that reduces potential loss of limited observations is a task of high importance. This section provides a review of recently developed optimization based methods for building optimal schedule of sensor surveillance.

2.3.1 Single Sensor Scheduling

The simplest case is to model one sensor that observes a group of sites at discrete time points. Some physical systems require virtually zero time for changing a site being observed. For example, the time of a camera refocusing, which is installed on a satellite, is negligibly small. This assumption leads to the model proposed by Yavuz and Jeffcoat in [46].

Assume that we need to observe n sites during T time periods. During every period a sensor is allowed to watch only at one of n sites. The scheduling decision can be modeled using binary variables $x_{i,t}$

$$x_{i,t} = \begin{cases} 1, & \text{if } i\text{-th site is observed at time } t, \\ 0, & \text{otherwise,} \end{cases} \tag{2.2}$$

t is a discrete variable and $t = 1, 2, ..., T$. If a site i is not observed for some period of time, it leads to the penalty that is proportional to the time of not observing this site. This penalty can be modeled using another group of decision variables. Let $y_{i,t}$ denotes the time of last visiting site i before time moment t. Let us note that variables $x_{i,t}$ completely determine values of $y_{i,t}$. Fixed penalty a_i and variable penalty b_{it} are associated with site i at time moment t. Thus, the penalty at time t associated with site i is

$$a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t}). \tag{2.3}$$

[46] suggests minimizing maximum loss over all sites and time intervals. Thus the objective function is defined as $\max_{i,t}\{a_i + b_{i,t}(t - y_{i,t})\}$. This objective function can be linearized and consequently the problem looks as following

$$\min C \tag{2.4}$$

$$\text{s.t.} \quad C \geq a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t}),\ \forall i = 1, \ldots, n,\ \forall t = 1, \ldots, T, \tag{2.5}$$

$$\sum_{i=1}^{n} x_{i,t} \leq 1,\ \forall t = 1, \ldots, T, \tag{2.6}$$

$$0 \leq y_{i,t} - y_{i,t-1} \leq t x_{i,t},\ \forall i = 1, \ldots, n,\ \forall t = 1, \ldots, T, \tag{2.7}$$

$$t x_{i,t} \leq y_{i,t} \leq t,\ \forall i = 1, \ldots, n,\ \forall t = 1, \ldots, T, \tag{2.8}$$

$$y_{i,0} = 0,\ \forall i = 1, \ldots, n, \tag{2.9}$$

$$x_{i,t} \in \{0, 1\},\ \forall i = 1, \ldots, n,\ \forall t = 1, \ldots, T, \tag{2.10}$$

$$y_{i,t} \in \mathbb{R}, \forall i = 1, \ldots, n,\ \forall t = 0, \ldots, T. \tag{2.11}$$

Constraints (3.9) ensure that the sensor visits only one site at a time. Constraints (3.10) -(3.11) set the dependence $y_{i,t}$ on $x_{i,t}$. That is $y_{i,t}$ is set to t if and only if the sensor is observing site i at time t otherwise $y_{i,t} = y_{i,t-1}$.

Single Sensor Scheduling problem is NP-hard [34] therefore various greedy heuristics have been proposed. The idea behind greedy algorithm is simple. At time $t = 1$ we find the site with the smallest penalty then at next time period we find another site with the smallest penalty. This sequence is repeated for all T time intervals. Thus the complexity of suggested approach is $O(nT)$. Yavuz and Jeffcoat has also suggested the "look ahead" modification of greedy heuristic which takes more computational time but computational experiment demonstrate that the solution is improved compared to the initial simple greedy optimization.

2.3.2 Stochastic approach

The stochastic nature of scheduling surveillance reduces the predictability of sensors behavior and therefore plays an important role for military tasks. Here we assume that sites are chosen randomly based on probability p_{ij} of transition from i-th to j-th site . Then sensor scheduling can be considered as a Markov chain stochastic process and characterized by steady state probabilities π_i.

The goal of stochastic approach is finding such steady state probabilities that minimize maximum loss. Let r_i be the visit period of site i. Then the penalty of information loss at site i is $a_i + (r_i - 1)b_{i,t}$ at time t. Let us consider a sufficiently small planning horizon with time-invariant site dynamics, this allows us to reduce $b_{i,t}$ to b_i and denote this approach as static. Visiting site i for every $r_i > 0$ periods is equal to spending $\pi_i = 1/r_i$ of the sensor's time at the site i. The optimal schedule is achieved when $\sum_i \pi_i = 1$; i.e., all the available time is utilized. Also, sensor never stays at any site for two consecutive periods of time. Thus $r_i \geq 2$ (or $\pi_i \leq 0.5$) should be satisfied for each site. Then the non-linear model for obtaining optimal stationary probabilities is formulated as

$$\min \max_i \{a_i + \left(\frac{1}{\pi_i} - 1\right) b_i\} \tag{2.12}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \pi_i = 1, \tag{2.13}$$

$$\pi_i \leq 0.5, \forall i = 1, \dots, n \tag{2.14}$$

$$\pi_i \in R, \forall i = 1, \dots, n \tag{2.15}$$

$$\tag{2.16}$$

A heuristic is proposed to solve this nonlinear continuous problem. Utilizing constraints (2.14), the authors define a lower bound on the objective function value with

$$C^L = \max_i \{a_i + b_i\}.$$

Then, we set

$$C = \max_i \{a_i + \left(\frac{1}{\pi} - 1\right) b_i\} = C^L$$

and calculate

$$r_i = (C - a_i)/b_i + 1$$

and

$$\pi_i = 1/r_i$$

for all i. Note that $\pi_i = 0.5$ for the sites with $a_i + b_i = C$ and $\pi_i < 0.5$ for the remaining constraints. If the determined probabilities add up to one then the optimal solution is found, and the procedure terminates. If the sum of probabilities is less than one, then one can shift some π-s up to make $\sum_i \pi_i = 1$. In the case when $\sum_i \pi_i > 1$ the found C is infeasible and we can apply iterative procedures (such as bisection) to find C such that $\sum_i \pi_i = 1$.

The static approach minimizes average penalty that is determined by steady probabilities and, thus, does not address the cases of long lasting absence at a site. Taking into account the fact that some random outcomes may result in extremely long penalties it is reasonable to increase the probabilities of visiting the sites that were visited long time ago. On the other hand, the probability of observing the sites, which were recently visited, should be decreased. Recall that $y_{i,t}$ represents the last time when the site i was visited by the time t. The probability of visiting a site must depend on the difference $t - y_{i,t}$, thus it will increase probability of visiting overdue sites.

To create a preference for visiting site i at time t the following adjustment factors are proposed

$$q_{i,t} = \pi_i \cdot \left(\frac{t - y_{i,t}}{r_i}\right)^k$$

It is bigger than one for overdue sites and less than one for the sites that have been visited within their expected visiting periods. The parameter k is a user-defined parameter, which determines the weight of the adjustment factor. The probabilities of visiting each site at specific time point t are based on previous history and can be computed as $p_i = q_i/Q$, where $Q = \sum_{i=1}^{n} q_i$.

Finally, a hybrid method was proposed based on a combination of greedy algorithm and the stochastic method, discussed above. The first step calculates the penalty of not visiting site i: $c_i = a_i + b_{i,t}(t - y_{i,t})$. Next step calculates preference values to visit each site: $q_i = \left(\frac{c_i}{c_{max}}\right)^k$, where $c_{max} = \max_i\{c_i\}$. And then the probabilities of visit are equal to: $p_i = q_i/Q$, where $Q = \sum_{i=1}^{n} q_i$.

2.3.3 Multiple Sensors Scheduling Using Percentile Type Constraints

Boyko *et al* have proposed to reformulate problem (3.32) -(3.35) for the case of m sensors [6].

$$\min \max_{i,t}\{a_i + b_{i,t}(t - y_{i,t})\} \tag{2.17}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_{i,t} \leq m, \ \forall t = 1, \ldots, T, \tag{2.18}$$

$$0 \leq y_{i,t} - y_{i,t-1} \leq t x_{i,t}, \ \forall i = 1, \ldots, n, \ \forall t = 1, \ldots, T, \tag{2.19}$$

$$t x_{i,t} \leq y_{i,t} \leq t, \ \forall i = 1, \ldots, n, \ \forall t = 1, \ldots, T, \tag{2.20}$$

$$y_{i,0} = 0, \ \forall i = 1, \ldots, n, \tag{2.21}$$

$$x_{i,t} \in \{0, 1\}, \ \forall i = 1, \ldots, n, \ \forall t = 1, \ldots, T, \tag{2.22}$$

$$y_{i,t} \in I\!R, \forall i = 1, \ldots, n, \ \forall t = 0, \ldots, T. \tag{2.23}$$

For every site i and every time point t we can calculate penalty associated with the last time a sensor visited this site $a_i + b_{i,t}(t - y_{i,t})$. Let us pick $(1 - \alpha)$ % of worst cases among these $n \times T$ penalty values. Then, instead of minimizing maximum loss we can minimize the the average taken over this $(1-\alpha)$ % percent of worst penalty values. Despite the fact that this formulation is deterministic it is equivalent to computing $(1 - \alpha)$ Conditional Value-at Risk (CVaR) for a set of random outcomes having equal probabilities $p_{i,t} = \frac{1}{nT}$.

CVaR [43, 45] is closely related to a well-known quantitative risk measure referred to as Value-at-Risk (VaR). By definition with respect to a specified probability level $(1 - \alpha)$ (in many applications the value of $(1 - \alpha)$ is set rather high, e.g. 95%), the α-VaR is the lowest amount ζ such that with probability $(1 - \alpha)$, the loss will not exceed ζ, whereas for continuous distributin the α-CVaR is the conditional expectation of losses *above* that amount ζ. As it can be seen, CVaR is a more conservative risk measure than VaR, which means that minimizing or restricting CVaR in optimization problems provides more robust solutions with respect to the risk of high losses (see figure 2.3).

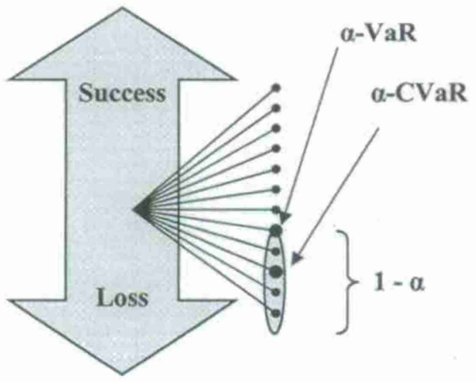

Figure 2.3: Graphical representation of VaR and CVaR.

The reader can find the formal definition of CVaR for various distribution cases in [43, 45]. Rockafellar and Uryasev [42, 28] showed that minimizing CVaR-type objective function of linear argument can be reduced to LP. In the sensors scheduling problem the loss function is introduced as

$$L(x, y, i, t) = a_i + b_{i,t}(t - y_{i,t}). \tag{2.24}$$

Thus the initial multiple sensors scheduling problem can be considered as the CVaR-based formulation

$$\min_{x,y} \mathrm{CVaR}_\alpha [L(x, y, i, t))] \tag{2.25}$$
$$\text{s.t.}$$
$$\text{constraints (2.18)-(2.23)},$$
$$\zeta \in I\!R. \tag{2.26}$$

Even though the parameters i and t of the loss function (2.24) are deterministic by nature, the stochastic framework was applied to this model in order to compute a percentile-type measure.

The parameters that quantify fixed and variable information losses are in many cases *uncertain* by nature. The following mathematical programming formulations allow quantifying and restricting the risks of worst-case losses associated with the aforementioned uncertain parameters. It is made by utilizing quantitative risk measures that allow one to control the robustness of the optimal solutions. Assuming that a_i and $b_{i,t}$ are random values we can formulate stochastic program utilizing the notion of CVaR.

$$\min_{x,y,\zeta} \mathrm{CVaR}_\alpha a_i^s(1 - x_{i,t}) + b_{i,t}^s(t - y_{i,t}) \tag{2.27}$$
$$\text{s.t.}$$
$$\text{constraints (2.18)-(2.23)}.$$

Here we use S scenarios (a_i^s and $b_{i,t}^s$, $s = 1, \ldots, S$) sampled from the distribution of penalty coefficients.

The mixed integer linear formulation was solved by ILOG CPLEX. The problems were also reformulated in terms of cardinality constraints and solved by AOrDa Portfolio Safeguard software package. Both solver find a good quality approximate solutions for large size problems.

2.4 Communication Network Interdiction

An important issue in military applications is to neutralize the communication in the sensors network of the enemy. This problem is known as jamming or eavesdropping a wireless communication network. This section introduces optimization formulations that allow to place jamming devices delivering maximal harm to the adverse sensors network. We start from a deterministic case when node locations are known.

The goal of jamming is to find a set of locations for placing jamming devices that suppresses the functionality of the network. Assume that n jamming devices are used to jam m communicating sensors. The assumption is made that the sensors and jammers can be located on a fixed set of locations V. The jamming effectiveness of device j is calculated as

$$d : (V \times V) \mapsto I\!R,$$

where d is a decreasing function of the distance from the jamming device to the node being jammed.

The cumulative level of jamming energy received at node i is defined as

$$Q_i := \sum_{j=1}^{n} d_{ij},$$

where n is the number of jamming devices. Then, jamming problem can be formulated as the minimization of the number of jamming devices placed, subject to a set of covering constraints:

$$\min n$$
$$\text{s.t.} Q_i \geq C_i, \quad i = 1, 2, \ldots, m.$$

Seeking the optimal placement coordinates $(x_j, y_j), j = 1, 2, \ldots, n$ for jamming devices given the coordinates $(X_i, Y_i), i = 1, 2, \ldots, m$ leads to non-convex formulations for most functions d. Thus, integer programming models for the problem are proposed.

A fixed set $\mathcal{N} = \{1, 2, \ldots, n\}$ of possible locations for the jamming devices and the set of communication nodes are introduced by Commander *et al* in [10]. Define the decision variable x_j as

$$x_j = \begin{cases} 1, & \text{if a jamming device is installed at location } j, \\ 0, & \text{otherwise.} \end{cases} \tag{2.28}$$

Then we have the optimal network covering formulation given as

$$\min \sum_{j=1}^{n} c_j x_j \tag{2.29}$$

$$\text{s.t.} \tag{2.30}$$

$$\sum_{j=1}^{n} d_{ij} x_j \geq C_i, \quad i = 1, 2, \ldots, m, \tag{2.31}$$

$$x_j \in \{0, 1\}, \quad j = 1, 2, \ldots, n, \tag{2.32}$$

Here the objective is to minimize the cost of jamming devices used while achieving some minimum level of coverage at each node. If $c_j = 1$ then the number of jammers is minimized.

If the goal is to suppress sensors communications we can minimize jamming cost with respect to the required level of connectivity index. Communication between nodes i and j is assumed to be destroyed if at least one of the nodes is jammed. Further, let $y_{ij} := 1$ if there exists a path from node i to node j in the jammed network and let $z_i = 1$ be an indicator that i-th node is jammed. This can be formulated as

$$\min \sum_{j=1}^{n} c_j x_j \tag{2.33}$$

$$\text{s.t.}$$

$$\sum_{\substack{j=1 \\ j \neq i}}^{m} y_{ij} \leq L, \ \forall\, i \in \mathcal{M}, \tag{2.34}$$

$$M(1 - z_i) > S_i - C_i \geq -M z_i, \ \forall\, i \in \mathcal{M}, \tag{2.35}$$

$$y_{ij} \text{ is consistent with the network and } z_i \tag{2.36}$$

$$x_j \in \{0, 1\}, \ \forall\, j \in \mathcal{N}, \tag{2.37}$$

$$z_i \in \{0, 1\}, \ \forall\, i \in \mathcal{M}, \tag{2.38}$$

$$y_{ij} \in \{0, 1\}, \ \forall\, i, j \in \mathcal{M}. \tag{2.39}$$

where $S_i := \sum_{j=1}^{n} d_{ij} x_j$ denote the cumulative level of jamming at node i, $M \in \mathbb{R}$ is some large constant. This problem can be formulated as a mixed integer linear problem and justification is provided in [10].

Finally, Commander *et al* provide percentile-based formulation for deterministic jamming problems. Suppose it is determined that jamming some fraction $\alpha \in (0, 1)$ of the nodes is sufficient for effectively dismantling the network. This can be accomplished by the inclusion of α-VaR constraints in the original model. Let $y : \mathcal{M} \mapsto \{0, 1\}$ be an indicator whether node i is jammed ($y_i = 1$).

Then to find the minimum number of jamming devices that will allow covering $\alpha \cdot 100\%$ of the network nodes with prescribed levels of jamming C_i, we must solve the following integer program

$$\min \sum_{j=1}^{n} c_j x_j \tag{2.40}$$

s.t.

$$\sum_{i=1}^{m} y_i \geq \alpha m, \tag{2.41}$$

$$\sum_{j=1}^{n} d_{ij} x_j \geq C_i y_i, \quad i = 1, 2, \ldots, m, \tag{2.42}$$

$$x_j \in \{0, 1\}, \quad j = 1, 2, \ldots, n, \tag{2.43}$$

$$y_i \in \{0, 1\}, \quad i = 1, 2, \ldots, m. \tag{2.44}$$

The α-CVaR optimization model for network covering can be formulated as a mixed integer linear program using a standard linearization framework:

$$\min \sum_{j=1}^{n} c_j x_j \tag{2.45}$$

s.t.

$$\zeta + \frac{1}{(1-\alpha)m} \sum_{i=1}^{m} \max\left\{ C_{\min} - \sum_{j=1}^{n} x_j d_{ij} - \zeta,\ 0 \right\} \leq 0, \tag{2.46}$$

$$\zeta \in \mathbb{R}, \tag{2.47}$$

$$x_j \in \{0, 1\}. \tag{2.48}$$

The VaR and CVaR models can also be written for connectivity suppression models in the similar fashion. We refer the reader to [10] for details.

The deterministic formulations of the wireless network jamming problem are extended in [9] to tackle the stochastic jamming problem formulations using percentile type constraints. These formulations consider the case when the exact topology of the network to be jammed is not known, but we know the distribution of network parameters.

Since the exact locations of the network nodes are unknown, it is assumed that a set of intelligence data has been collected and from that a set $\mathcal{S}$ of the most likely scenarios have been compiled. Scenario $s \in \mathcal{S}$ contains both the node locations $\{(\xi_1^s, \eta_1^s), (\xi_2^s, \eta_2^s), \ldots, (\xi_m^s, \eta_m^s)\}$ and the set of jamming thresholds $\{C_1^s, C_2^s, \ldots, C_m^s\}$. For each scenario $s \in \mathcal{S}$, the set $\mathcal{M}_s = \{1, 2, \ldots, m_s\}$ needs to be jammed. Taking into account all the scenarios we can write a mathematical program for node covering problem:

$$\min \sum_{k=1}^{n} c_k x_k, \tag{2.49}$$

s.t.

$$\sum_{k=1}^{n} d_{ik}^s x_k \geq C_i^s,\ i = 1, 2, \ldots, m_s, s = 1, 2, \ldots, S, \tag{2.50}$$

$$x_k \in \{0, 1\},\ k = 1, 2, \ldots, n, \tag{2.51}$$

It is unlikely to find the solution that can provide effective jamming strategy for all scenarios. Therefore, the notion of percentile-based risk measures can be utilized to develop formulations of the robust jamming problems incorporating these risk constraints. The robust node covering problem with Value-at-Risk constraints can be formulated as

$$\min \sum_{k=1}^{n} c_k x_k, \tag{2.52}$$

s.t.

$$\sum_{k=1}^{n} d_{ik}^s x_k \geq C_i^s \rho_i^s, \; \forall\, s \in \mathcal{S}, \forall\, i \in \mathcal{M}_s, \tag{2.53}$$

$$\sum_{i=1}^{m_s} \rho_i^s \geq \alpha m_s, \; \forall\, s \in \mathcal{S}, \tag{2.54}$$

$$x_k \in \{0,1\}, \; \forall\, k \in \mathcal{N}, \tag{2.55}$$

$$\rho_i^s \in \{0,1\}, \; \forall\, s \in \mathcal{S}, \forall\, i \in \mathcal{M}_s, \tag{2.56}$$

The loss function can be considered as the difference between the energy required to jam network node i, namely C_i^s, and the cumulative amount of energy received at node i due to x over each scenario. With this the robust node covering problem with CVaR constraints is formulated as follows.

$$\min \sum_{k=1}^{n} c_k x_k, \tag{2.57}$$

s.t.

$$\zeta^s + \frac{1}{(1-\alpha)m_s} \tag{2.58}$$

$$\sum_{i=1}^{m_s} \max\left\{ C_{min}^s - \sum_{k=1}^{n} d_{ik}^s x_k - \zeta^s, 0 \right\} \leq 0, \; \forall\, s \in \mathcal{S}, \tag{2.59}$$

$$x_k \in \{0,1\}, \; \forall\, k \in \mathcal{N}, \tag{2.60}$$

$$\zeta^s \in \mathbb{R}, \; \forall\, s \in \mathcal{S}. \tag{2.61}$$

The CVaR constraint (2.59) implies that for the $(1-\alpha)\cdot 100\%$ of the worst (least) covered nodes, the average value of $f(x)$ is less than or equal to 0.

Numerical experiments demonstrate that problems of moderate size can be solved by ILOG CPLEX. Finding effective heuristics and approximations for sensors network jamming problems of large dimensions is an interesting and important research direction.

2.5 Conclusion

In this chapter, we presented a brief outline of problems and challenges arising in the important and exciting area of mathematical programming techniques for sensor network applications. Clearly, the research in this area is far from complete, and a lot of extensions and generalizations of the presented models can be developed. Overall, the main challenges arising in this area are associated with efficiently incorporating the uncertainties in the mathematical programming formulations and dealing with the computational intractability of the corresponding stochastic and robust optimization problems. However, there are promising developments in this field, including computationally efficient heuristic algorithms and software packages for solving these problems. Therefore, the area of sensor networks and mathematical programming techniques associated with it has a clear potential from both theoretical and practical perspectives.

Chapter 3

Robust Connectivity Issues in Dynamic Sensor Networks for Area Surveillance under Uncertainty Problems

3.1 Introduction

In this chapter, we address several problems and challenges arising in the task of utilizing dynamic sensor networks for area surveillance. This task needs to be efficiently performed in different applications, where various types of information need to be collected from multiple locations. In addition to obtaining potentially valuable information (that can often be time-sensitive), one also needs to ensure that the information can be efficiently transmitted between the nodes in a wireless communication/sensor network. In the simplest static case, the location of sensors (i.e., nodes in a sensor network) is fixed, and the links (edges in a sensor network) are determined by the distance between sensor nodes, that is, two nodes would be connected if they are located within their wireless transmission range. However, in many practical situations, the sensors are installed on moving vehicles (for instance, unmanned air vehicles (UAVs)) that can dynamically move within a specified area of surveillance. Clearly, in this case the location of nodes and edges in a network and the overall network topology can change significantly over time. The task of crucial importance in these settings is to develop optimal strategies for these dynamic sensor networks to operate efficiently in terms of both collecting valuable information and ensuring robust wireless connectivity between sensor nodes.

In terms of collecting information from different locations (sites), one needs to deal with the challenge that the number of sites that need to be visited to gather potentially valuable information is usually much larger than the number of sensors. Under these conditions, one needs to develop efficient *schedules* for all the moving sensors such that the amount of valuable information collected by the sensors is maximized. A relevant approach that was previously used by the co-authors to address this challenge dealt with formulating this problem in terms of minimizing the *information losses* due to the fact that some locations are not under surveillance at certain time moments. In these settings, the information losses can be quantified as both *fixed* and *variable* losses, where fixed losses would occur when a given site is simply not under surveillance at some time moment, while variable losses would increase with time depending on how long a site has not been visited by a sensor. Taking into account variable losses of information is often critical in the cases of dealing with strategically important sites that need to be monitored as closely as possible. In addition, the parameters that quantify fixed and variable information losses are usually *uncertain*, therefore, the uncertainty and risk should be explicitly addressed in the corresponding optimization problems.

The other important challenge that will be addressed in this paper is ensuring *robust connectivity patterns* in dynamic sensor networks. These robustness properties are especially important in uncertain and adverse environments in military settings, where uncertain failures of network components (nodes and/or edges) can occur.

The considered robust connectivity characteristics will deal with different parameters of the network. First, the nodes within a network should be connected by paths that are not excessively long, that is, the number of intermediary nodes and edges in the information transmission path should be small enough. Second, each node should be connected to a significant number of other nodes in a network, which would provide the possibility of multiple (backup) transmission paths in the network, since otherwise the network topology would be vulnerable to possible network component failures.

Clearly, the aforementioned robust connectivity properties are satisfied if there are direct links between all pairs of nodes, that is, if the network forms a *clique*. Cliques are very robust network structures, due to the fact that they can sustain multiple network component failures. Note that any subgraph of a clique is also a clique, which implies that this structure would maintain robust connectivity patterns even if multiple nodes in the network are disabled. However, the practical drawbacks of cliques include the fact that these structures are often overly restrictive and expensive to construct.

To provide a tradeoff between robustness and practical feasibility, certain other network structures that "relax" the definition of a clique can be utilized. The following definitions address these relaxations from different perspectives. Given a graph $G(V, E)$ with a set of vertices (nodes) V and a set of edges E, a k-clique C is a set of vertices in which any two vertices are distance at most k from each other in G [40]. Let $d_G(i, j)$ be the length of a shortest path between vertices i and j in G and $d(G) = \max\limits_{i,j \in V} d_G(i, j)$ be the *diameter* of G.

Thus, if two vertices $u, v \in V$ belong to a k-clique C, then $d_G(u, v) \leq k$, however this does not imply that $d_{G(C)}(u, v) \leq k$ (that is, other nodes in the shortest path between u and v are *not required to belong to the k-clique*). This motivated Mokken [41] to introduce the concept of a *k-club*. A *k-club* is a subset of vertices $D \subseteq V$ such that the *diameter of induced subgraph* $G(D)$ is at most k (that is, there exists a path of length at most k connecting any pair of nodes within a k-club, where *all the nodes* in this path also belong to this k-club). Also, $\tilde{V} \subseteq V$ is said to be a k-plex if the degree of every vertex in the induced subgraph $G(\tilde{V})$ is at least $|\tilde{V}| - k$ [44]. A comprehensive study of the maximum k-plex problem is presented in a recent work by Balasundaram et al. [38].

In this chapter, we utilize these concepts to develop rigorous mathematical programming formulations to model robust connectivity structures in dynamic sensor networks. Moreover, these formulations will also take into account various uncertain parameters by introducing quantitative risk measures that minimize or restrict information losses. Overall, we will develop optimal "schedules" for sensor movements that will take into account both the uncertain losses of information and the robust connectivity between the nodes that would allow one to efficiently exchange the collected information.

3.2 Multi-Sensor Scheduling Problems: General Deterministic Setup

This section introduces a preliminary mathematical framework for dynamic multi-sensor scheduling problems. The simplest deterministic one-sensor version of this problem was introduced in [46]. The one-sensor scheduling problem was then extended and generalized to more realistic cases of multi-sensor scheduling problems, including the setups in uncertain environments by Boyko et al. [39]. In the subsequent sections of this paper, this setup will be further extended to incorporate robust connectivity issues into the considered dynamic sensor network models.

To facilitate further discussion, we first introduce the following mathematical notations that will be used throughout the paper. Assume that there are m sensors that can move within a specified area of surveillance, and there are n sites that need to be observed at every discrete time moment $t = 1, \ldots, T$. One can initially assume that a sensor can observe only one site at one point of time and can immediately switch to another site at the next time moment. Since m is usually significantly smaller than n, there will be "breaches" in surveillance that can cause losses of potentially valuable information.

A possible objective that arises in practical situations is to build a strategy that optimizes a potential loss that is associated with not observing certain sites at some time moments.

One can introduce binary decision variables

$$x_{i,t} = \begin{cases} 1, & \text{if } i\text{-th site is observed at time } t, \\ 0, & \text{otherwise,} \end{cases} \tag{3.1}$$

and integer variables $y_{i,t}$ that denote the last time site i was visited as of the end of time t, $i = 1, \ldots, n$, $t = 1, \ldots, T$, $m < n$.

One can then associate a fixed penalty a_i with each site i and a variable penalty b_i of information loss. If a sensor is away from site i at time point t, the fixed penalty a_i is incurred. Moreover, the variable penalty b_i is proportional to the time interval when the site is not observed. We assume that the variable penalty rate can be dynamic; therefore, the values of b_i may be different at each time interval. Thus the loss at time t associated with site i is

$$a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t}). \tag{3.2}$$

In the considered setup, we want to minimize the *maximum penalty* over all time points t and sites i

$$\max_{i,t}\{a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})\}. \tag{3.3}$$

Furthermore, $x_{i,t}$ and $y_{i,t}$ are related via the following set of constraints. No more than m sensors are used at each time point; therefore

$$\sum_{i=1}^{n} x_{i,t} \le m,\ \forall t = 1, \dots, T. \tag{3.4}$$

Time $y_{i,t}$ is equal to the time when the site i was last visited by a sensor by time t. This condition is set by the following constraints:

$$0 \le y_{i,t} - y_{i,t-1} \le t x_{i,t},\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3.5}$$

$$t x_{i,t} \le y_{i,t} \le t,\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3.6}$$

Further, using an extra variable C and standard linearization techniques, we can formulate the multi-sensor scheduling optimization problem in the deterministic setup as the following mixed integer linear program:

$$\min C \tag{3.7}$$

$$\text{s.t.}\quad C \ge a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t}),\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3.8}$$

$$\sum_{i=1}^{n} x_{i,t} \le m,\ \forall t = 1, \dots, T, \tag{3.9}$$

$$0 \le y_{i,t} - y_{i,t-1} \le t x_{i,t},\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3.10}$$

$$t x_{i,t} \le y_{i,t} \le t,\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3.11}$$

$$y_{i,0} = 0,\ \forall i = 1, \dots, n, \tag{3.12}$$

$$x_{i,t} \in \{0, 1\},\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3.13}$$

$$y_{i,t} \in \mathbb{R}, \forall i = 1, \dots, n,\ \forall t = 0, \dots, T. \tag{3.14}$$

We allowed relaxation (3.14) of variables $y_{i,t}$ to the space of real numbers, because the constraints (3.5) and (3.6) enforce the feasible values of variables $y_{i,t}$ to be integer.

3.2.1 Cardinality Formulation

Lemma. Constraint (3.13) is equivalent to the following combination of two constraints:

$$0 \le x_{i,t} \le 1\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T\ , \tag{3.15}$$

$$\text{card}(\mathbf{x}_t) \le \sum_{i=1}^{n} x_{i,t}\ \forall t = 1, \dots, T\ , \tag{3.16}$$

where $\mathbf{x}_t = (x_{1,t}, \dots, x_{n,t})^T$, and card$(\mathbf{x}_t)$ denotes the cardinality function for the vector $\mathbf{x}_t$. By definition, card$(\mathbf{x}_t)$ equals the number of non-zero elements in the input vector $\mathbf{x}_t$.
Proof. Assume the matrix $(x_{i,t})$ satisfies constraint (3.13). Obviously, it then satisfies (3.15). At the same time, for every t, sum of all elements is equal to the number of values 1 in it. And these are the only non-zero elements in it. Therefore, constraint (3.16) is also satisfied.

Now assume the matrix $(x_{i,t})$ does not satisfy constraint (3.13). Thus there is a pair (i_δ, t_δ), for which $x_{i_\delta,t_\delta} = \delta$ and $\delta \neq 0$ and $\delta \neq 1$. If $\delta < 0$ or $\delta > 1$, then constraint (3.15) is violated. Thus, for all pairs (i,t), $0 \leq x_{i,t} \leq 1$, and $0 < \delta < 1$. Therefore, for all pairs (i,t), $\text{card}(x_{i,t}) \geq x_{i,t}$, and $\text{card}(\delta) > \delta$. Taking into account that $\text{card}(\mathbf{x}_t) = \sum_i \text{card}(x_{i,t})$ we conclude that (3.16) is violated. $\diamond$

Now we can write alternative, cardinality formulation for the general deterministic sensor-scheduling problem.

$$\min C \tag{3.17}$$

$$\text{s.t.} \quad C \geq a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t}),\ \forall i = 1,\ldots,n,\ \forall t = 1,\ldots,T, \tag{3.18}$$

$$\sum_{i=1}^{n} x_{i,t} \leq m,\ \forall t = 1,\ldots,T, \tag{3.19}$$

$$0 \leq y_{i,t} - y_{i,t-1} \leq t x_{i,t},\ \forall i = 1,\ldots,n,\ \forall t = 1,\ldots,T, \tag{3.20}$$

$$t x_{i,t} \leq y_{i,t} \leq t,\ \forall i = 1,\ldots,n,\ \forall t = 1,\ldots,T, \tag{3.21}$$

$$y_{i,0} = 0,\ \forall i = 1,\ldots,n, \tag{3.22}$$

$$0 \leq x_{i,t} \leq 1,\ \forall i = 1,\ldots,n,\ \forall t = 1,\ldots,T, \tag{3.23}$$

$$\text{card}(\mathbf{x}_t) \leq \sum_{i=1}^{n} x_{i,t},\ \forall t = 1,\ldots,T\ , \tag{3.24}$$

$$y_{i,t} \in I\!R, \forall i = 1,\ldots,n,\ \forall t = 0,\ldots,T. \tag{3.25}$$

Although the two formulations are equivalent, some optimization solvers, such as Portfolio Safeguard (that will be mentioned later in this paper), can provide a near-optimal solution faster if the formulation with cardinality constraints is used instead of the one with boolean variables, which may be important in time-critical systems in military settings.

3.3 Quantitative Risk Measures in Uncertain Environments: Conditional Value-at-Risk

To facilitate further discussion on the formulations of the aforementioned problems under uncertainty, in this section we briefly review basic definitions and facts related to the Conditional Value-at-Risk concept.

Conditional Value-at-Risk (CVaR) [42, 43, 45] is a quantitative risk measure that will be used in the models developed in the next section, which will take into account the presence of uncertain parameters. CVaR is closely related to a well-known quantitative risk measure referred to as Value-at-Risk (VaR). By definition, with respect to a specified probability level α (in many applications the value of α is set rather high, e.g. 95%), the α-VaR is the lowest amount η_α such that with probability α, the loss will not exceed η_α, whereas for continuous distributions the α-CVaR is the conditional expectation of losses *above* that amount η_α. As it can be seen, CVaR is a more conservative risk measure than VaR, which means that minimizing or restricting CVaR in optimization problems provides more robust solutions with respect to the risk of high losses.

Formally, α-CVaR for continuous distributions can be expressed as

$$\text{CVaR}_\alpha(\mathbf{x}) \ = \ (1-\alpha)^{-1} \int_{L(\mathbf{x},\mathbf{w}) \geq \eta_\alpha(\mathbf{x})} L(\mathbf{x},\mathbf{w})\, p(\mathbf{w})\, d\mathbf{w}, \tag{3.26}$$

where $L(\mathbf{x},\mathbf{w})$ is the random loss (penalty) variable driven by decision vector $\mathbf{x}$ and having a distribution in $I\!R$ induced by that of the vector of uncertain parameters $\mathbf{w}$.

CVaR is defined in a similar way for discrete or mixed distributions. The reader can find the formal definition of CVaR for general case in [43, 45].

It has been shown by Rockafellar and Uryasev [42] that minimizing (3.26) is equivalent to minimizing the function

$$F_\alpha(\mathbf{x},\eta) \ = \ \eta + (1-\alpha)^{-1} \int_{\mathbf{w} \in I\!R^d} [L(\mathbf{x},\mathbf{w}) - \eta]^+\, p(\mathbf{w})\, d\mathbf{w} \tag{3.27}$$

over w and η, where $[t]^+ = t$ when $t > 0$ but $[t]^+ = 0$ when $t \leq 0$, and optimal value of the variable η corresponds to the VaR value η_α, introduced above.

3.4 Optimizing the Connectivity of Dynamic Sensor Networks Under Uncertainty

This section extends the previous sensors scheduling problem to a stochastic environment. We use CVaR measure to model and optimize various objectives associated with the risk of loss of information.

In the stochastic formulation, the penalties a_i and $b_{i,t}$ are random. We generate S discrete scenarios, which approximate implied joint distribution. Thus, every scenario consists of two arrays: one-dimensional $\{a_i\}^s$ and two-dimensional $\{b_{i,t}\}^s$.

Now, consider the term of the loss function corresponding to the site i, time t, and scenario s:

$$L^s(x, y; i, t) = a_i^s(1 - x_{i,t}) + b_{i,t}^s(t - y_{i,t}).$$

Under uncertainty, it is often more important to mitigate the biggest possible losses, rather than the average damage. Following this idea, we take $(1 - \alpha)$ biggest penalties, and minimize average penalty over all i, t and s. This objective function is exactly the conditional value-at-risk.

We now have the following class of optimization problems:

$$\min_{x,y} CVaR_\alpha\{L(x, y; i, t)\} \tag{3.28}$$

This class has one extreme case: $\alpha = 1$, when the problem becomes equivalent to minimizing maximum possible penalty over all scenarios, locations and time points:

$$\min_{x,y} \max_{i,t,s}(a_i^s(1 - x_{i,t}) + b_{i,t}^s(t - y_{i,t})). \tag{3.29}$$

This problem has an equivalent LP formulation:

$$\min C \tag{3.30}$$

$$\text{s.t.} \quad C \geq a_i^s(1 - x_{i,t}) + b_{i,t}^s(t - y_{i,t}), \tag{3.31}$$

$$\forall i = 1, \ldots, n, \forall t = 1, \ldots, T, \forall s = 1, \ldots, S.$$

In order to formulate a general CVaR optimization problem in LP terms we have to introduce additional variables $\tau_{i,t}^s$, $s = 1, \ldots, S$, $i = 1, \ldots, n$, $t = 1, \ldots, T$, and η. With these variables the problem of minimizing CVaR will be reduced to the following:

$$\min C \tag{3.32}$$

$$\text{s.t.} \quad C \geq \eta + \frac{1}{(1-\alpha)nST} \sum_{\substack{s=1,\ldots,S \\ i=1,\ldots,n \\ t=1,\ldots,T}} \tau_{i,t}^s, \tag{3.33}$$

$$\tau_{i,t}^s \geq a_i^s(1 - x_{i,t}) + b_{i,t}^s(t - y_{i,t}) - \eta, \tag{3.34}$$

$$\forall i = 1, \ldots, n,\ t = 1, \ldots, T,\ s = 1, \ldots, S,$$

$$\tau_{i,t}^s \geq 0,\ \forall i = 1, \ldots, n,\ t = 1, \ldots, T,\ s = 1, \ldots, S. \tag{3.35}$$

We have discussed various objective functions with objective-specific constraints for sensor scheduling problems in the stochastic environment. In addition to that, every sensor scheduling problem, including those in stochastic environment, must have constraints limiting number of sensors (3.4) and defining variables of the last time of observation (3.5)– (3.6). These constraints are referred to as mandatory constraints for every sensor-scheduling problem.

Further we define a wireless connectivity network $G(V, E)$ on the set of locations V. We interpret it in terms of the 0-1 adjacency matrix $E = \{e_{ij}\}_{i,j=1,\ldots,n}$, where each e_{ij} is a 0-1 indicator of wireless signal reachability between nodes i and j, that is, if locations i and j are within direct transmission distance from each other, then they are connected by an edge, and $e_{ij} = 1$ ($e_{ij} = 0$ otherwise). We also define a subnetwork $\tilde{G}$ of $G(V, E)$ containing only those m nodes (locations) that are directly observed by sensors at a particular time moment.

Scheduling of observation often requires sensors to maintain a certain level of wireless connectivity robustness. If an enemy sends a jamming signal that breaks connectivity between a pair of nodes, then the subnetwork $\tilde{G}$ either should stay connected, or at least should maintain unity with probability close to 1. Further, we will utilize several types of network structures that can be applied to ensure that the network satisfies certain robustness constraints.

The most robust network structure is a clique, which implies that each pair of nodes is directly connected by an edge. Obviously, maintaining a clique structure of the subnetwork $\tilde{G}$ at every moment in time is very expensive in terms of penalty, and can be even impossible, if the overall wireless connectivity network is not dense enough. Hence, it is reasonable to utilize appropriate types of *clique relaxations* to ensure robust network connectivity at every time moment.

One of the considered concepts is a k-plex. By definition, as mentioned above, a k-plex is a subgraph in which every node is connected to at least $m - k$ other nodes in it (where m is the number of nodes in this subgraph). This network configuration ensures that each node is connected to multiple neighbors, which makes it more challenging for an adversary to disconnect the network and isolate the nodes by destroying (jamming) the edges.

Another considered class of network configurations is a k-club. Recall that every pair of nodes in k-club is connected in it through a chain of no more than k arcs (edges). The motivation for studying this type of constraints is based on the fact that if two sensors are connected through a shorter path, it lowers the probability of errors in information transmission through intermediaries, since the number of intermediaries is smaller. Later in the paper, we will specifically use a stronger requirement on the length of these paths. We require that any two nodes are connected either directly by an edge, or through *at most one* intermediary node, which is often a desired robustness requirement under the conditions when the number of intermediary information transmissions needs to be minimized due to adversarial conditions. Clearly, a *2-club* is a structure that satisfies this requirement. In the next subsection, we show that this condition can be incorporated in the considered optimization models.

3.4.1 Ensuring Short Transmission Paths via 2-club Formulation

The general requirement for a subnetwork $\tilde{G}$ to represent a k-club can be formulated as the following set of constraints:

$$\begin{aligned}
& e_{ij} + \ldots \\
& \quad + \sum_{q=1}^{n} e_{iq} e_{qj} x_{q,t} + \ldots \\
& \quad + \sum_{q=1}^{n} \sum_{l=1}^{n} e_{iq} e_{ql} e_{lj} x_{q,t} x_{l,t} + \ldots \\
& \quad + \sum_{q=1}^{n} \sum_{l=1}^{n} \sum_{p=1}^{n} e_{iq} e_{ql} e_{lp} e_{pj} x_{q,t} x_{l,t} x_{p,t} + \ldots \\
& \quad + \sum_{i_1=1}^{n} \sum_{i_2=1}^{n} \cdots \sum_{i_{k-2}=1}^{n} \sum_{i_{k-1}=1}^{n} e_{ii_1} e_{i_1 i_2} \ldots e_{i_{k-2} i_{k-1}} e_{i_{k-1} j} x_{i_1,t} \ldots x_{i_{k-1},t} \geq \\
& \geq x_{i,t} + x_{j,t} - 1,
\end{aligned} \tag{3.36}$$

where $i = 1, \ldots, n-1,\ j = i+1, \ldots, n,\ t = 1, \ldots, T$. For every k these constraints can be linearized, however, the size of the problem may substantially increase. In this chapter, we limit our discussion only to 2-club constraints due to the practical reasons mentioned earlier in this section and due to the fact that the formulation for the case of $k = 2$ will not add too many new entities (no more than $O(n^2)$) to the problem formulation. They require every pair of nodes (i, j) to be connected directly, or through some other node p. Such type of communication between sensors

(i, j) has a concise formulation:

$$e_{ij} + \sum_{p=1}^{n} e_{ip}e_{pj}x_{p,t} \geq x_{i,t} + x_{j,t} - 1 \; \forall i = 1, \ldots, n-1, \; \forall j = i+1, \ldots, n,$$
$$\forall t = 1, \ldots, T.$$

Here, the left-hand side is always nonnegative. The right-hand side becomes positive only if both locations i and j are observed by sensors, and then it equals 1. According to the 2-club definition, these sensors have to be connected (and exchange information) either directly, or through one other intermediary sensor node. In the first case e_{ij} equals 1. In the second case, the sum $\sum_{p=1}^{n} e_{ip}e_{pj}x_{p,t}$ will also be positive.

It is also important to note that those constraints, for which $e_{ij} = 1$, can be omitted. Thus, a 2-club wireless network configuration can be ensured by the following set of constraints:

$$\sum_{p \in \delta(i) \cap \delta(j)} x_{p,t} \geq x_{i,t} + x_{j,t} - 1,$$
$$\forall i = 1, \ldots, n-1, \; \forall j = i+1, \ldots, n, \; j \notin \delta(i), \; \forall t = 1, \ldots, T,$$

where $\delta(i)$ and $\delta(j)$ are the sets of neighbors of nodes i and j, respectively.

Below we present the complete general formulation for the dynamic sensor scheduling optimization problem in a stochastic environment with 2-club wireless connectivity constraints.

$$\begin{aligned}
& \min C \\
\text{s.t.} \quad & C \geq \eta + \frac{1}{(1-\alpha)nST} \sum_{\substack{s=1,\ldots,S \\ i=1,\ldots,n \\ t=1,\ldots,T}} \tau_{i,t}^{s} \\
& \tau_{i,t}^{s} \geq a_{i}^{s}(1 - x_{i,t}) + b_{i,t}^{s}(t - y_{i,t}) - \eta \\
& \forall i = 1, \ldots, n, \; t = 1, \ldots, T, \; s = 1, \ldots, S, \\
& \tau_{i,t}^{s} \geq 0, \; \forall i = 1, \ldots, n, \; t = 1, \ldots, T, \; s = 1, \ldots, S, \\
& \sum_{i=1}^{n} x_{i,t} \leq m, \; \forall t = 1, \ldots, T, \\
& 0 \leq y_{i,t} - y_{i,t-1} \leq t x_{i,t}, \; \forall i = 1, \ldots, n, \; \forall t = 1, \ldots, T, \\
& t x_{i,t} \leq y_{i,t} \leq t, \; \forall i = 1, \ldots, n, \; \forall t = 1, \ldots, T, \\
& y_{i,0} = 0, \; \forall i = 1, \ldots, n, \\
& \sum_{p \in \delta(i) \cap \delta(j)} x_{p,t} \geq x_{i,t} + x_{j,t} - 1 \\
& \forall i = 1, \ldots, n-1, \; \forall j = i+1, \ldots, n, \; j \notin \delta(i), \; \forall t = 1, \ldots, T, \\
& x_{i,t} \in \{0, 1\}, \; \forall i = 1, \ldots, n, \; \forall t = 1, \ldots, T, \\
& y_{i,t} \in I\!R, \; \forall i = 1, \ldots, n, \; \forall t = 0, \ldots, T.
\end{aligned}$$

3.4.2 Ensuring Backup Connections via k-plex Formulation

Constraints that require a wireless network to have the k-plex structure, can be defined using a symmetric adjacency matrix $E = \{e_{ij}\}_{i,j=1,\ldots,n}$, as defined above. Recall that $\mathbf{x}_t = (x_{1,t}, \ldots, x_{n,t})^T$. Consider the vector $\mathbf{z}_t = (z_{1,t}, \ldots, z_{n,t}) = E\mathbf{x}_t$. The element $z_{i,t}$ can be interpreted as the number of sensors which have a wireless connection with node i at time t. Thus, the constraint $E\mathbf{x}_t \geq \mathbf{x}_t$ or $(E - I)\mathbf{x}_t \geq 0$ ensures that each sensor node has at least one neighbor, i.e., it is not isolated. If we want each sensor to have at least $(m-k)$ wireless connections (edges) with other sensors, then we should make the constraints more restrictive: $E\mathbf{x}_t \geq (m-k)\mathbf{x}_t$, or

$$(E-(m-k)I)\,\mathbf{x}_t \geq 0 \;\forall t = 1,\dots,T.$$

These restrictions by definition ensure that a subnetwork $\tilde{G}$ is a k-plex.

Below we present the complete general formulation for the dynamic sensor scheduling optimization problem in a stochastic environment with k-plex wireless connectivity constraints.

$$\begin{aligned}
&\min C\\
\text{s.t.}\quad & C \geq \eta + \frac{1}{(1-\alpha)nST}\sum_{\substack{s=1,\dots,S\\ i=1,\dots,n\\ t=1,\dots,T}} \tau_{i,t}^{s}\\
& \tau_{i,t}^{s} \geq a_i^s(1-x_{i,t}) + b_{i,t}^{s}(t-y_{i,t}) - \eta\\
& \forall i = 1,\dots,n,\ t=1,\dots,T,\ s=1,\dots,S,\\
& \tau_{i,t}^{s} \geq 0,\ \forall i=1,\dots,n,\ t=1,\dots,T,\ s=1,\dots,S,\\
& \sum_{i=1}^{n} x_{i,t} \leq m,\ \forall t=1,\dots,T,\\
& 0 \leq y_{i,t} - y_{i,t-1} \leq t x_{i,t},\ \forall i=1,\dots,n,\ \forall t=1,\dots,T,\\
& t x_{i,t} \leq y_{i,t} \leq t,\ \forall i=1,\dots,n,\ \forall t=1,\dots,T,\\
& y_{i,0}=0,\ \forall i=1,\dots,n,\\
& (E-(m-k)I)\,\mathbf{x}_t \geq 0,\ \forall t=1,\dots,T\\
& x_{i,t}\in\{0,1\},\ \forall i=1,\dots,n,\ \forall t=1,\dots,T,\\
& y_{i,t}\in I\!R,\ \forall i=1,\dots,n,\ \forall t=0,\dots,T.
\end{aligned}$$

3.5 Computational Experiments

Computational experiments on sample problem instances have been performed on Intel Xeon X5355 2.66 GHz CPU with 16GB RAM, using two commercial optimization solvers: ILOG CPLEX 11.2 and AORDA PSG 64 bit (MATLAB 64 bit environment). It should be noted that due to the nature of the considered class of problems, they are computationally challenging even on relatively small networks. Therefore, in many practical situations, finding near-optimal solutions in a reasonable time would be sufficient. The PSG package was used in addition to CPLEX because it has attractive features in terms of coding the optimization problems, and therefore it may be more preferable to use in practical time-critical settings. In particular, in addition to linear and polynomial functions, PSG supports a number of different classes of functions, such as CVaR and cardinality functions. For the purposes of the current case study we defined in PSG the objective using the CVaR function, and we also used cardinality function for the cardinality constraint on $x_{i,t}$ instead of boolean constraint.

For comparison purposes, multiple experiments have been performed. All experiments were divided into two groups: with 2-club connectivity constraints on subnetwork $\tilde{G}$, and k-plex constraints with $k = \left[\frac{m}{2}\right]$. In each of these groups, number of locations $n = 10, 11, 12, 13, 14, 15$ and number of sensors $m = 4, 5, 6, 7, 8$. All problems have CVaR-type objective with $\alpha = 0.9$, deterministic setup (1 scenario), 20 time intervals. The edge density of the considered overall wireless connectivity network was $\rho = 0.8$ (80% pairs of nodes are connected).

We have run PSG using two built-in solvers: CAR and TANK. These solvers took on average 26 seconds to deliver solution over all cases with 2-club constraints, and 27 seconds for the cases with k-plex constraints. After that we run CPLEX on cases with 2-club constraints with time limit 26 seconds, on cases with k-plex constraints with time limit 27 seconds. Then, we additionally run CPLEX on all cases with time limit 1 minute. Computational results are presented in two tables, for the cases with 2-club constraints and k-plex constraints, respectively.

Table 3.1: CPLEX Results: Problem with 2-club Constraints

Case		PSG CAR		PSG TANK		CPLEX 26 sec		CPLEX 1 min	
n	m	value	time	value	time	value	time	value	time
10	4	97.96	22.3	100.61	21.6	84.62	26.1	84.62	60.1
10	5	79.29	22.8	83.69	25.6	74.30	26.0	71.57	60.0
10	6	74.79	24.2	71.15	23.6	63.32	26.1	63.32	60.1
10	7	64.82	25.6	61.68	26.5	57.54	26.1	57.54	60.0
10	8	52.27	26.7	51.86	25.2	50.76	26.1	50.21	60.0
11	4	106.19	23.2	105.16	24.2	92.21	26.1	92.21	60.1
11	5	86.61	23.0	85.21	22.4	75.54	26.1	75.54	60.1
11	6	75.08	23.9	76.21	22.5	70.79	26.1	66.32	60.1
11	7	68.43	25.4	69.93	24.0	58.37	26.0	58.37	60.1
11	8	60.44	24.4	61.05	23.8	57.01	26.1	57.01	60.1
12	4	122.32	24.1	124.00	22.8	105.45	26.1	105.45	60.1
12	5	91.25	22.8	98.02	22.6	82.08	26.2	81.96	60.1
12	6	84.69	22.8	81.76	23.0	72.65	26.0	72.65	60.1
12	7	73.44	23.6	76.95	22.1	64.11	26.1	64.11	60.1
12	8	64.78	25.6	61.95	24.0	56.10	26.1	56.10	60.1
13	4	126.37	24.5	119.76	28.9	98.46	26.0	98.46	60.1
13	5	94.48	24.0	104.78	26.5	86.31	26.0	88.20	60.1
13	6	82.46	24.2	83.29	27.2	76.61	26.1	76.61	60.1
13	7	73.97	25.5	74.59	30.8	70.53	26.1	67.93	60.0
13	8	71.57	26.4	69.79	33.9	59.92	26.1	59.92	60.1
14	4	135.75	25.6	139.06	27.3	118.41	26.0	112.74	60.1
14	5	109.75	27.1	114.27	27.2	95.01	26.1	94.87	60.1
14	6	89.58	24.3	93.82	27.2	79.54	26.1	79.54	60.1
14	7	80.70	25.9	80.89	23.3	70.53	26.2	70.37	60.1
14	8	75.31	26.0	76.88	26.6	65.26	26.1	61.67	60.1
15	4	155.67	27.9	145.00	26.7	127.00	26.2	126.82	60.1
15	5	113.18	25.8	115.65	28.8	104.06	26.1	102.34	60.1
15	6	95.51	24.8	99.11	28.1	90.80	26.1	82.74	60.1
15	7	85.96	25.0	86.49	27.8	74.22	26.1	74.22	60.1
15	8	77.81	26.1	76.83	34.4	68.09	26.1	68.22	60.0
			avg		avg		avg		avg
			24.8		26.0		26.1		60.1

The results show that on average the best solution is produced by CPLEX 1 minute run. Values, obtained by CPLEX runs with 26 and 27 seconds limits are by 1.2% and 2.2% greater for 2-club and for k-plex respectively. In most cases solutions obtained by two runs were equal. Therefore, CPLEX obtains solution close to optimal in about less than 30 seconds. PSG TANK solution value is greater than CPLEX 1 minute solution value by 15.8% and 22.4% for 2-club and for k-plex respectively. PSG CAR performs slightly better than another solver, providing the solution values greater than CPLEX 1 minute solution values by 15.0% and 22.0%.

In addition to deterministic setup, we have run the aforementioned optimization problems under uncertainty on several stochastic problem instances with the number of sensors $m = 6$, the number of locations $n = 12$, the CVaR-type objective with $\alpha = 0.9$, $T = 10$ time intervals, for different numbers of scenarios: $S = 10, 20, 50, 100$. As before, the wireless connectivity network edge density was $\rho = 0.8$. The time limit was set to 5 minutes. PSG solvers in most cases provided solution before the time limit was reached. However, the quality of solution was worse then provided by CPLEX by 15% on average.

Table 3.2: CPLEX Results: Problem with k-plex Constraints

Case		PSG CAR		PSG TANK		CPLEX 27 sec		CPLEX 1 min	
n	m	value	time	value	time	value	time	value	time
10	4	106.52	25.1	102.94	22.7	85.34	27.1	85.34	60.1
10	5	79.98	23.7	83.48	22.0	72.40	27.1	72.40	60.1
10	6	74.22	25.0	78.25	26.0	63.22	27.0	63.22	60.1
10	7	65.71	27.0	62.27	26.9	56.73	27.0	56.73	60.0
10	8	55.70	25.8	50.91	24.3	50.26	27.0	50.26	60.0
11	4	117.61	23.7	101.28	26.8	87.57	27.0	87.57	60.1
11	5	89.16	24.7	92.77	22.9	75.97	27.1	75.97	60.0
11	6	77.27	24.3	84.08	26.2	68.25	27.1	67.03	60.0
11	7	70.4	26.0	67.24	23.5	58.61	27.1	58.61	60.0
11	8	67.98	25.9	64.84	23.4	53.80	27.1	53.15	60.0
12	4	134.62	31.1	128.60	26.1	109.89	27.1	102.52	60.1
12	5	100.80	24.4	103.70	23.0	80.11	27.1	80.11	60.1
12	6	83.21	24.3	87.56	25.5	70.54	27.1	70.54	60.1
12	7	69.91	25.8	74.99	26.2	63.26	27.1	63.26	60.1
12	8	70.03	27.9	69.28	22.7	56.75	27.1	56.73	60.1
13	4	134.39	32.2	121.72	28.5	103.87	27.1	103.87	60.1
13	5	97.24	25.4	103.89	23.7	84.63	27.1	84.57	60.1
13	6	90.51	25.7	89.40	29.5	77.94	27.1	77.94	60.1
13	7	78.15	26.2	77.34	28.7	68.52	27.1	68.52	60.1
13	8	77.49	26.7	72.61	24.3	63.36	27.1	59.69	60.1
14	4	134.01	30.3	140.12	32.1	119.17	27.1	112.18	60.1
14	5	114.72	26.9	113.61	27.6	90.00	27.1	89.34	60.1
14	6	97.83	27.4	96.71	29.1	78.37	27.1	78.37	60.1
14	7	86.09	26.4	87.00	31.4	70.46	27.1	70.24	60.1
14	8	77.78	27.2	75.75	26.0	62.48	27.0	62.48	60.1
15	4	153.18	34.4	189.41	32.3	136.89	27.1	120.81	60.1
15	5	123.61	28.6	123.84	27.0	110.94	27.1	98.15	60.2
15	6	97.53	27.2	101.72	31.4	83.11	27.0	82.59	60.1
15	7	93.21	27.3	86.48	30.1	74.43	27.1	74.43	60.0
15	8	80.29	28.9	75.70	25.7	67.08	27.1	67.37	60.1
			avg		avg		avg		avg
			26.9		26.5		27.1		60.1

Table 3.3: CPLEX and PSG Results: Stochastic Setup

		CPLEX			PSG CAR		PSG TANK	
type	S	value	time	gap	value	time	value	time
k-plex	10	85.14	300.1	27.0%	96.78	25.3	96.81	29.4
k-plex	20	89.36	300.2	34.6%	95.10	35.3	96.10	37.1
k-plex	50	92.27	300.5	44.4%	110.06	154.7	97.40	273.1
k-plex	100	93.81	301.7	49.7%	104.57	300.6	115.69	300.6
2-club	10	86.92	300.1	30.5%	100.13	229.9	100.13	300.1
2-club	20	84.92	300.1	32.0%	97.55	79.4	97.55	300.1
2-club	50	89.75	300.5	44.3%	104.30	300.1	103.42	300.1
2-club	100	95.87	301.8	50.7%	116.63	300.2	116.23	300.2

3.6 Conclusions

We have defined and extended a class of dynamic sensor scheduling problems by introducing explicit robust connectivity requirements, specifically, k-club and k-plex constraints, taking into account wireless connectivity requirements for sensors at every time moment.

We have also presented computational results for moderate-size instances in both deterministic and stochastic problem setups. Since the size of the stochastic version of the problem is S times larger than for the deterministic version (where S is the number of implied penalty scenarios), solving these stochastic problems is clearly challenging from the computational perspective.

Problem formulations can be further extended by adding movement network and corresponding constraints as introduced in Boyko et al. [39], thus modeling the map of possible sensor movements.

We have also compared the performance of the two optimization software packages: ILOG CPLEX 11.2 and AORDA Portfolio Safeguard 64 bit. Although CPLEX generally outperforms PSG on the considered problem instances in terms of the quality of solutions by 15–22%, PSG may have its own advantages in practical settings due to a more user-friendly approach and more options to formulate a problem.

The classes of problems considered in this paper are primarily motivated by military applications; however, the developed formulations are general enough so that they can be applied in a variety of settings.

Bibliography

[1] American optimal decision, portfolio safeguard. *http://www.aorda.com*.

[2] Ilog cplex. *http://www.ilog.com/products/cplex/*.

[3] Ameer Ahmed Abbasi and Mohamed Younis. A survey on clustering algorithms for wireless sensor networks. *Comput. Commun.*, 30(14-15):2826–2841, 2007.

[4] Ian F. Akyildiz, Tommaso Melodia, and Kaushik R. Chowdhury. A survey on wireless multimedia sensor networks. *Comput. Netw.*, 51(4):921–960, 2007.

[5] Pratik Biswas and Yinyu Ye. Semidefinite programming for ad hoc wireless sensor network localization. In *IPSN '04: Proceedings of the 3rd international symposium on Information processing in sensor networks*, pages 46–54, New York, NY, USA, 2004. ACM.

[6] Nikita Boyko, Timofey Turko, Vladimir Boginski, David Jeffcoat, Stanislav Uryasev, Panos Pardalos, and Greg Zrazhevsky. Robust multi-sensor scheduling for multi-site surveillance. *Journal of Combinatorial Optimization*, submitted.

[7] Anna L. Buczak, Henry (Hui) Wang, Houshang Darabi, and Mohsen A. Jafari. Genetic algorithm convergence study for sensor network optimization. *Inf. Sci. Inf. Comput. Sci.*, 133(3-4):267–282, 2001.

[8] Amit S. Chhetri, Darryl Morrell, and Antonia Papandreou-Suppappola. Nonmyopic sensor scheduling and its efficient implementation for target tracking applications. *EURASIP J. Appl. Signal Process.*, 2006(1):9–9, uary.

[9] C.W. Commander, P.M. Pardalos, V. Ryabchenko, S. Sarykalin, T. Turko, and S. Uryasev. Robust wireless network jamming problems. In C.W. Commander, M.J. Hirsch, R.A. Murphey, and P.M. Pardalos, editors, *Lecture Notes in Control and Information Sciences*. Springer, 2008.

[10] C.W. Commander, P.M. Pardalos, V. Ryabchenko, and S. Uryasev. The wireless network jamming problem. *Journal of Combinatorial Optimization*, 14:4:481–498, 2007.

[11] Konstantinos P. Ferentinos and Theodore A. Tsiligiridis. Adaptive design optimization of wireless sensor networks using genetic algorithms. *Comput. Netw.*, 51(4):1031–1051, 2007.

[12] Matthias Hollick, Ivan Martinovic, Tronje Krop, and Ivica Rimac. A survey on dependable routing in sensor networks, ad hoc networks, and cellular networks. In *EUROMICRO '04: Proceedings of the 30th EUROMICRO Conference*, pages 495–502, Washington, DC, USA, 2004. IEEE Computer Society.

[13] Jaehoon Jeong, Sarah Sharafkandi, and David H. C. Du. Energy-aware scheduling with quality of surveillance guarantee in wireless sensor networks. In *DIWANS '06: Proceedings of the 2006 workshop on Dependability issues in wireless ad hoc networks and sensor networks*, pages 55–64, New York, NY, USA, 2006. ACM.

[14] Andreas Klappenecker, Hyunyoung Lee, and Jennifer L. Welch. Scheduling sensors by tilinglattices. In *PODC '08: Proceedings of the twenty-seventh ACM symposium on Principles of distributed computing*, pages 437–437, New York, NY, USA, 2008. ACM.

[15] Dimitrios Koutsonikolas, Saumitra M. Das, and Y. Charlie Hu. Path planning of mobile landmarks for localization in wireless sensor networks. *Comput. Commun.*, 30(13):2577–2592, 2007.

[16] Mauri Kuorilehto, Marko Hännikäinen, and Timo D. Hämäläinen. A survey of application distribution in wireless sensor networks. *EURASIP J. Wirel. Commun. Netw.*, 5(5):774–788, 2005.

[17] Y. Li, M. T. Thai, , and W. Wu (eds). *Wireless Sensor Networks and Applications*. Springer, 2007.

[18] M. S. Lobo, M. Fazel, and S. Boyd. Portfolio optimization with linear and fixed transaction costs. *Annals of Operations Research*, 152(1):376–394, 2007.

[19] Dragos Niculescu and Badri Nath. Ad hoc positioning system (aps) using aoa. In *The 28th Conference on Computer Communications*, pages 1734–1743. IEEE, 2003.

[20] P.M. Pardalos, Y. Ye, and C.W. Commander (eds) V. Boginski. *Sensors: Theory, Algorithms, and Applications*. Springer, to appear in 2009.

[21] Joseph C. Pemberton and III Flavius Galiber. A constraint-based approach to satellite scheduling. In *DIMACS workshop on on Constraint programming and large scale discrete optimization*, pages 101–114, Boston, MA, USA, 2001. American Mathematical Society.

[22] R.T. Rockafellar and S.P. Uryasev. Optimization of conditional value-at-risk. *Journal of Risk*, 2:21–42, 2000.

[23] R.T. Rockafellar and S.P. Uryasev. Conditional value-at-risk for general loss distributions. *Journal of Banking and Finance*, 26:1443–1471, 2002.

[24] Masoomeh Rudafshani and Suprakash Datta. Localization in wireless sensor networks. In *IPSN '07: Proceedings of the 6th international conference on Information processing in sensor networks*, pages 51–60, New York, NY, USA, 2007. ACM.

[25] Sergey Sarykalin, Gaia Serraino, and Stanislav Uryasev. Var vs cvar in risk management and optimization. *INFORMS Tutorial*, 2008.

[26] Sumeetpal S. Singh, Nikolaos Kantas, Ba-Ngu Vo, Arnaud Doucet, and Robin J. Evans. Simulation-based optimal sensor scheduling with application to observer trajectory planning. *Automatica*, 43(5):817–830, 2007.

[27] Anthony Man-Cho So and Yinyu Ye. Theory of semidefinite programming for sensor network localization. *Math. Program.*, 109(2):367–384, 2007.

[28] S. Uryasev. Conditional value-at-risk: Optimization algorithms and applications. *Financial Engineering News*, 14:1–5, 2000.

[29] Swaroop Venkatesh and R. Michael Buehrer. A linear programming approach to nlos error mitigation in sensor networks. In *IPSN '06: Proceedings of the 5th international conference on Information processing in sensor networks*, pages 301–308, New York, NY, USA, 2006. ACM.

[30] Chen Wang and Li Xiao. Sensor localization in concave environments. *ACM Trans. Sen. Netw.*, 4(1):1–31, 2008.

[31] Kui Wu, Yong Gao, Fulu Li, and Yang Xiao. Lightweight deployment-aware scheduling for wireless sensor networks. *Mob. Netw. Appl.*, 10(6):837–852, 2005.

[32] Kui Wu, Chong Liu, Jianping Pan, and Dandan Huang. Robust range-free localization in wireless sensor networks. *Mob. Netw. Appl.*, 12(5):392–405, 2007.

[33] Ting Yan, Yu Gu, Tian He, and John A. Stankovic. Design and optimization of distributed sensing coverage in wireless sensor networks. *Trans. on Embedded Computing Sys.*, 7(3):1–40, 2008.

[34] M. Yavuz and D.E. Jeffcoat. An analysis and solution of the sensor scheduling problem. In *Advances in Cooperative Control and Optimization*, volume 369, pages 167–177. Springer, 2007.

[35] M. Yavuz and D.E. Jeffcoat. Single sensor scheduling for multi-site surveillance. Technical report, Air Force Research Laboratory, 2007.

[36] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. Wireless sensor network survey. *Comput. Netw.*, 52(12):2292–2330, 2008.

[37] Yong Yuan, Zongkai Yang, Min Chen, and Jianhua He. A survey on information processing technologies in wireless sensor networks. *Int. J. Ad Hoc Ubiquitous Comput.*, 1(3):103–109, 2006.

[38] B. Balasundaram, S. Butenko, and I. Hicks. Clique Relaxations in Social Network Analysis: The Maximum k-plex Problem, *Operations Research*, to appear, 2010.

[39] N. Boyko, T. Turko, V. Boginski, D.E. Jeffcoat, S. Uryasev, G. Zrazhevsky, P.M. Pardalos. Robust Multi-Sensor Scheduling for Multi-Site Surveillance. *Journal of Combinatorial Optimization*, in press, 2009.

[40] R.D. Luce. Connectivity and generalized cliques in sociometric group structure. *Psychometrika*, **15**:169–190, 1950.

[41] R.J. Mokken. Cliques, clubs and clans. *Quality and Quantity*, **13**:161–173, 1979.

[42] R.T. Rockafellar and S.P. Uryasev. Optimization of conditional value-at-risk. *Journal of Risk*, 2:21-42, 2000.

[43] R.T. Rockafellar and S.P. Uryasev. Conditional value-at-risk for general loss distributions. *Journal of Banking and Finance*, 26:1443-1471, 2002.

[44] S. B. Seidman and B. L. Foster. A graph theoretic generalization of the clique concept. *Journal of Mathematical Sociology*, **6**:139–154, 1978.

[45] S. Sarykalin, G. Serraino, and S. Uryasev. VaR vs CVaR in risk management and optimization. *INFORMS Tutorial*, 2008.

[46] M. Yavuz and D.E. Jeffcoat. Single sensor scheduling for multi-site surveillance. Technical report, Air Force Research Laboratory, 2007.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE FINAL REPORT	3. DATES COVERED *(From - To)* 01 Mar 08 – 30 Nov 10

4. TITLE AND SUBTITLE Detecting and Jamming Dynamic Communication Networks in Ati-Access Environments	5a. CONTRACT NUMBER
	5b. GRANT NUMBER FA9550-08-1-0190
	5c. PROGRAM ELEMENT NUMBER 61102F
6. AUTHOR(S) Prof Stanislav Uryasev Prof Panos Pardalos	5d. PROJECT NUMBER 2304
	5e. TASK NUMBER DX
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) University of Florida Dept of Industrial & System Eng. 474 Weil Hall Gainesville, FL 32611-6550	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Office of Scientific Research 875 North Randolph Street Suite 325, Room 3112 Arlington, VA 22203-1768	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-OSR-VA-TR-2012-0072

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approve for Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Three chapters present methods for handling various optimization problems related to sensors networks. The first chapter investigates mathematical programming techniques for solving a class of multi-sensor scheduling problems. We conducted several case studies and investigated the performance of robust optimization solvers for considered 0-1 linear programming problems. The second chapter presents a survey describing recent developments in the area of mathematical programming techniques for various types of sensor network applications. The corresponding mathematical programming formulations and solving methods are discussed. The third chapter considers several classes of problems that deal with optimizing the performance of dynamic sensor networks used for area surveillance under uncertainty with robust radio connectivity constraints. Various formulations of optimization problems and computational results are presented

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(Include area code)*